**UNIVERSIDADE TUIUTI DO PARANÁ**

**ADRIANO SILVA VIANNA**

**CHRISTIAN CARLOS WEBER DE OLIVEIRA**

**JOSÉ MARIO CORREÂ SANTOS**

**LUIZ HENRIQUE BLIND LUSTOSA**

**MATHEUS AFORNALI**

**BONJOU: SISTEMA DE CARPOOLING PARA IMIGRANTES**

**CURITIBA**

**2017**

**ADRIANO SILVA VIANNA**

**CHRISTIAN CARLOS WEBER DE OLIVEIRA**

**JOSÉ MARIO CORREÂ SANTOS**

**LUIZ HENRIQUE BLIND LUSTOSA**

**MATHEUS AFORNALI**

**BONJOU: SISTEMA DE CARPOOLING PARA IMIGRANTES**

Trabalho de Conclusão de Curso elaborado como requisito parcial para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas da Universidade Tuiuti do Paraná.

Orientador: Prof. Sérgio Luiz M. Filho

**CURITIBA**

**2017**

**RESUMO**

A partir da crescente onda de imigrantes, os quais se viram obrigados a deixar suas casas em busca de um lar mais seguro, este projeto apresenta uma proposta de sistema *carpooling* ou carona solidária. Neste projeto está descrita a documentação relacionada ao desenvolvimento do sistema *Bonjou*, projeto criado para aproximar imigrantes e nativos por meio de um sistema de carona solidária, em parceria com a Organização não governamental *O planeta é um só*. A documentação referente ao portal *Bonjou* se resume em requisitos funcionais e não funcionais do sistema, os casos de uso, o diagrama de classe e de sequência, bem como o banco de dados e o *script*.

**Palavras-chave:** Sistema, Carona solidária, Bonjou, O planeta é um só.

**LISTA DE FIGURAS**

[FIGURA 1 – DIAGRAMA GERAL DE CASO DE USO 21](#_Toc498949118)

[FIGURA 2 – DIAGRAMA DE CASO DE USO CARONISTA 22](#_Toc498949119)

[FIGURA 3 – DIAGRAMA DE CASO DE USO CARONEIRO 23](#_Toc498949120)

[FIGURA 4 – DIAGRAMA DE CASO DE USO DO SISTEMA 23](#_Toc498949121)

[FIGURA 5 - DIAGRAMA DE CLASSE 28](#_Toc498949122)

[FIGURA 6 - FLUXO DE TELAS 29](#_Toc498949123)

[FIGURA 7 - DIAGRAMA DE SEQUENCIA I 30](#_Toc498949124)

[FIGURA 8 - DIAGRAMA DE SEQUÊNCIA II 31](#_Toc498949125)

[FIGURA 9 - DIAGRAMA SEQUENCIA III 31](#_Toc498949126)

[FIGURA 10 - DIAGRAMA DE SEQUÊNCIA IV 32](#_Toc498949127)

[FIGURA 11 - DIAGRAMA DE SEQUÊNCIA V 33](#_Toc498949128)

[FIGURA 12 - DIAGRAMA DE SEQUÊNCIA VI 33](#_Toc498949129)

[FIGURA 13 – MODELO CONCEITUAL DO BANCO DE DADOS 34](#_Toc498949130)

**LISTA DE QUADROS**

[QUADRO 1 – TIPO DO REQUISITO 13](#_Toc498883441)

[QUADRO 2 – PRIORIDADE DO REQUISITO 13](#_Toc498883442)

[QUADRO 3 – REQUISITO FUNCIONAL I 14](#_Toc498883443)

[QUADRO 4 – REQUISITO FUNCIONAL II 14](#_Toc498883444)

[QUADRO 5 – REQUISITO FUNCIONAL III 14](#_Toc498883445)

[QUADRO 6 – REQUISITO FUNCIONAL IV 15](#_Toc498883446)

[QUADRO 7 – REQUISITO FUNCIONAL V 15](#_Toc498883447)

[QUADRO 8 – REQUISITO FUNCIONAL VI 15](#_Toc498883448)

[QUADRO 9 – REQUISITO FUNCIONAL VII 16](#_Toc498883449)

[QUADRO 10 – REQUISITO FUNCIONAL XIII 16](#_Toc498883450)

[QUADRO 11 – REQUISITO FUNCIONAL XIX 17](#_Toc498883451)

[QUADRO 12 – REQUISITO FUNCIONAL X 17](#_Toc498883452)

[QUADRO 13 – REQUISITO FUNCIONAL XI 17](#_Toc498883453)

[QUADRO 14 - REQUISITO FUNCIONAL XII 18](#_Toc498883454)

[QUADRO 15 - REQUISITO FUNCIONAL XIII 18](#_Toc498883455)

[QUADRO 16 - REQUISITO FUNCIONAL XIV 18](#_Toc498883456)

[QUADRO 17 - REQUISITO FUNCIONAL XV 19](#_Toc498883457)

[QUADRO 18 - REQUISITO FUNCIONAL XVI 19](#_Toc498883458)

[QUADRO 19 - REQUISITO NÃO FUNCIONAL I 19](#_Toc498883459)

[QUADRO 20 - REQUISITO NÃO FUNCIONAL II 20](#_Toc498883460)

[QUADRO 21 - REQUISITO NÃO FUNCIONAL III 20](#_Toc498883461)

[QUADRO 22 - REQUISITO NÃO FUNCIONAL IV 20](#_Toc498883462)

[QUADRO 23 - REQUISITO NÃO FUNCIONAL V 20](#_Toc498883463)

[QUADRO 24 - REQUISITO NÃO FUNCIONAL VI 20](#_Toc498883464)

[QUADRO 25 – ESPECIFICAÇÃO I 24](#_Toc498883465)

[QUADRO 26 – ESPECIFICAÇÃO II 24](#_Toc498883466)

[QUADRO 27 – ESPECIFICAÇÃO III 24](#_Toc498883467)

[QUADRO 28 – ESPECIFICAÇÃO IV 25](#_Toc498883468)

[QUADRO 29 – ESPECIFICAÇÃO V 25](#_Toc498883469)

[QUADRO 30 – ESPECIFICAÇÃO VI 25](#_Toc498883470)

[QUADRO 31 – ESPECIFICAÇÃO VIII 26](#_Toc498883471)

[QUADRO 32 – ESPECIFICAÇÃO IX 26](#_Toc498883472)

[QUADRO 34 - ESPECIFICAÇÃO X 26](#_Toc498883473)

[QUADRO 35 - ESPECIFICAÇÃO XI 27](#_Toc498883474)

**SUMÁRIO**

[1 INTRODUÇÃO 11](#_Toc498965094)

[2 VISÃO GERAL DO SISTEMA 12](#_Toc498965095)

[2.1 FERRAMENTAS ADICIONAIS 12](#_Toc498965096)

[3 REQUISITOS DO SISTEMA 13](#_Toc498965097)

[3.1 REQUISITOS FUNCIONAIS 13](#_Toc498965098)

[3.2 REQUISITOS NÃO FUNCIONAIS 19](#_Toc498965099)

[4 CASO DE USO 21](#_Toc498965100)

[4.1 ESPECIFICAÇÃO DE CASO DE USO 24](#_Toc498965101)

[5 DIAGRAMA DE CLASSES 28](#_Toc498965102)

[6 DIAGRAMA DE TELAS 29](#_Toc498965103)

[7 DIAGRAMA DE SEQUÊNCIA 30](#_Toc498965104)

[8 BANCO DE DADOS 34](#_Toc498965105)

[9 SCRIPTS 35](#_Toc498965106)

[10 CONCLUSÃO 47](#_Toc498965107)

[REFERÊNCIAS 48](#_Toc498965108)

[APÊNDICE I – Relatório de Testes 49](#_Toc498965109)

[APÊNDICE II – Telas 54](#_Toc498965110)

# INTRODUÇÃO

De acordo com uma pesquisa da Organização das Nações Unidas (ONU), o número de imigrantes ao redor do mundo alcançou 244 milhões em 2015, dentre eles, o número de refugiados é de 20 milhões (UNITED NATIONS, 2016). Diante dessa realidade, diversos imigrantes deslocando-se ao redor do globo, faz-se necessário pensar soluções que permitam esses cidadãos de exercerem livremente seus papéis sociais e profissionais. Nesse sentido, este trabalho apresenta um sistema online de *carpooling* para imigrantes recém-chegados ao Brasil.

O *carpooling* é uma plataforma online de carona solidária que permite o uso compartilhado de um automóvel particular por duas ou mais pessoas. O objetivo é economizar despesas de transporte, bem como contribuir para a redução do congestionamento e a poluição do ar. O diferencial do *Bonjou[[1]](#footnote-1)*, sistema desenvolvido para este Trabalho de Conclusão de Curso, é reduzir os obstáculos encontrados pelos imigrantes ao chegarem ao Brasil.

Sabe-se que, ao desembarcar em um novo país, o imigrante encontra dificuldades em relação ao idioma, à cultura e também à locomoção da cidade. Nesse sentido, o objetivo do *Bonjou* é reduzir as fronteiras do imigrante para uma vida mais independente. Essa redução ocorre por meio do sistema de *carpooling* que aproxima o recém-chegado a um morador nativo, o qual pode ajudar essa parcela da população a deslocar-se pela cidade, principalmente no que se refere a trabalho e estudos, sem ônus para o imigrante, de modo colaborativo.

Tendo essas perspectivas em mente, nos próximos capítulos estão descritos os requisitos do sistema, o caso de uso, o diagrama de classes e de sequência, o banco de dados e, por fim, os *scripts* utilizados para o desenvolvimento e criação do sistema. Os produtos gerados com o desenvolvimento do projeto são: Plataforma Web, documentação, diagramas, relatório de teste e manual do usuário.

# VISÃO GERAL DO SISTEMA

Neste capítulo estão descritos os métodos utilizados para a construção do sistema proposto, desde o padrão de arquitetura empregado, além de *APIs*, *frameworks* e outras ferramentas.

Para a realização do projeto foi utilizado o padrão MVC – *Model*, *View* e *Controller*, um dos padrões de arquitetura mais antigos e utilizados no mundo. O Padrão MVC consiste em dividir o sistema em partes e gerencia-los através dos objetos Modelo, Visão e Controle (*Model*, *View* e *Controller* – respectivamente).

A respeito do objeto Modelo (*Model*), é a classe que se comunica com o banco de dados. A parte referente à Visão (*View*) é a camada de interação com o usuário e exibição dos dados. Já o objeto controle (*Controller*) é responsável pelas requisições do usuário.

## FERRAMENTAS ADICIONAIS

Como já mencionado, o sistema utiliza como base o *framework* Laravel, que aplica a arquitetura MVC e tem como objetivo criar aplicações sólidas com alto nível de segurança e desempenho. Emprega também a linguagem PHP, que atua no lado do servidor, para conexão com banco de dados.

Outras linguagens aplicadas ao sistema são o HTML (*Hypertext Markup Language*) e o Javascript, sendo o primeiro uma linguagem de marcação para estruturar páginas web, e a segunda linguagem que manipula o HTML e o CSS oferecendo uma página mais dinâmica ao usuário, atuando no lado do cliente.

Para colocar no ar o portal foi hospedado no serviço Umbler que utiliza um servidor GNU/Linux com a distribuição Ubuntu. O domínio bonju.com.br foi registrado no site registro.br.

# REQUISITOS DO SISTEMA

Os requisitos do sistema são descritos de forma acessível para que o usuário sem os conhecimentos técnicos possa entender como o sistema funciona. Além disso, são subdivididos em duas categorias: funcionais e não funcionais. O primeiro descreve os comportamentos do produto e o segundo descreve as condições e a qualidade para que o produto seja feito de forma correta. Os quadros a seguir estão ordenados em requisitos funcionais e não funcionais, além de mostrar dois quadros contendo as legendas que auxiliam a compreensão dos demais quadros.

Quadro 1 – Tipo do requisito

|  |  |
| --- | --- |
| **Código** | **Descrição** |
| E | Externo |
| O | Organizacional |
| P | Produto |

Fonte: os autores (2017)

Quadro 2 – Prioridade do requisito

|  |  |
| --- | --- |
| **Código** | **Urgência** |
| 1 | Baixa |
| 2 | Média |
| 3 | Alta |

Fonte: os autores (2017)

## REQUISITOS FUNCIONAIS

Requisitos funcionais constituem as funcionalidades e os serviços do sistema. É o que o sistema faz. Em seguida serão mostrados os quadros com os requisitos do portal Bonjou.

QUADRO 3 – REQUISITO FUNCIONAL I

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RF01 | Manter *Login* | P | 3 | RF02 |
| **Descrição** | O usuário ou administrador fazem *login* no site | | | |
| **Especificação** | 1. **Fluxo Básico:**    1. O usuário acessa a *url* do sistema (www.bonjou.com.br);    2. O usuário seleciona o botão login;    3. O usuário preenche os campos com seu e-mail e senha;    4. O sistema verifica os dados e acessa o sistema; 2. **Fluxo Alternativo:**    1. Usuário não possui cadastro;    2. Clica no botão “Cadastrar”    3. O sistema redireciona para a tela de cadastro;   3.1 O usuário não lembra a senha;  3.2 O usuário clica no botão “Esqueci minha senha”; | | | |

Fonte: os autores (2017)

QUADRO 4 – REQUISITO FUNCIONAL II

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RF02 | Manter Cadastro | P | 2 | -- |
| **Descrição** | O usuário faz cadastro no site | | | |
| **Especificação** | 1. **Fluxo Básico:**    1. O usuário acessa a *url* do sistema (www.bonjou.com.br);    2. Clica no botão “Cadastrar-se”    3. O usuário entra com seus dados (Apelido, e-mail, senha, confirmar senha);    4. O usuário confirma se os dados foram adicionados corretamente;    5. Clica no botão Cadastrar. 2. **Fluxo Alternativo:**   1.1 O usuário clica no botão de *login* e acessa a tela de login. | | | |

Fonte: os autores (2017)

QUADRO 5 – REQUISITO FUNCIONAL III

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RF03 | Manter Avaliação | P | 1 | RF01 |
| **Descrição** | O usuário mantém uma avaliação do motorista/carona | | | |
| **Especificação** | **1 Fluxo Básico**  1.1 O usuário mantém avaliação da carona e do caroneiro;  1.2 E avalia a carona e o caroneiro.  **2 Fluxo Alternativo**  2.1 O usuário exclui a avaliação;  2.2 O usuário altera a avaliação. | | | |

Fonte: os autores (2017)

QUADRO 6 – REQUISITO FUNCIONAL IV

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RF04 | Manter Perfil | P | 2 | RF01 |
| **Descrição** | O usuário mantém seu cadastro atualizado. | | | |
| **Especificação** | **1 Fluxo Básico**   * 1. O usuário clica no botão “Meu perfil”   2. Verifica as informações, e clica em “editar”;   3. E atualiza os seguintes dados:   - Nome  - Sobrenome  - Idade  - Gênero  - Sobre mim  - Telefone  - Foto (Imagem)  1.4 E clica no botão “Atualizar Perfil”;  **2 Fluxo Alternativo**  2.1 Caso o usuário não tenha perfil o sistema redireciona para a tela de Cadastro de Perfil; | | | |

Fonte: os autores (2017)

QUADRO 7 – REQUISITO FUNCIONAL V

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RF05 | Solicitar Carona | P | 3 | RF01 |
| **Descrição** | O usuário solicita carona | | | |
| **Especificação** | **1 Fluxo Básico**   * 1. O usuário clica no botão “Caronas”;   2. Usuário visualiza a lista de caronas disponíveis;   3. Clica sobre a viagem escolhida;   4. O usuário confirma carona.   **2 Fluxo Alternativo**  2.1 O usuário clica no botão “Cancelar carona”;  2.2 O usuário clica no botão “Alterar carona”; | | | |

Fonte: os autores (2017)

QUADRO 8 – REQUISITO FUNCIONAL VI

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RF06 | Manter carona | P | 2 | RF01 |
| **Descrição** | O usuário mantém uma carona | | | |
| **Especificação** | **1 Fluxo Básico**   * 1. O usuário mantém a solicitação da carona;   2. E pode consultar os agendamentos ativos.   **2 Fluxo Alternativo**  2.1 O usuário clica em alterar carona;  2.2 O usuário clica em excluir carona. | | | |

Fonte: os autores (2017)

QUADRO 9 – REQUISITO FUNCIONAL VII

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RF07 | Oferecer Carona | P | 1 | RF08 |
| **Descrição** | O usuário cadastra uma oferta de carona | | | |
| **Especificação** | **1 Fluxo Básico**   * 1. O usuário clica no botão “Oferecer carona”;   2. O usuário preenche os campos, com os dados:   - Local de saída;  - Local de destino;  - Data de partida;  - Horário de partida;  - Veículo;  - Número de passageiros;  1.3 Clica no botão enviar.  **2 Fluxo Alternativo**   * 1. O usuário excluí a carona cadastrada;   2. O usuário clica no botão traçar rota, visualizando o resumo da viagem através do mapa;   **3 Fluxo Exceção**  3.1 Se o usuário não tem um veículo cadastrado, ele será direcionado a tela de “Meus veículos”; | | | |

Fonte: os autores (2017)

QUADRO 10 – REQUISITO FUNCIONAL XIII

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RF08 | Manter Veículo | P | 1 | RF01 |
| **Descrição** | Manter cadastro de veículo | | | |
| **Especificação** | **1 Fluxo Básico**   * 1. Com o perfil logado, o usuário clica no botão Meus veículos;   2. Usuário clica no botão simbolizado pelo “+”;   3. O usuário entra com os dados de seu veículo:   - Marca  - Modelo  - Ano  - Cor  - Placa  - Número de passageiros   * 1. Verifica se os dados estão corretos;   2. Clica no botão Salvar Veículo.   **2 Fluxo Alternativo**  2.1 O usuário clica no botão “Excluir veículo”;  2.2 O usuário clica no botão “Alterar veículo”. | | | |

Fonte: os autores (2017)

QUADRO 11 – REQUISITO FUNCIONAL XIX

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RF09 | Notificar solicitação de carona | P | 2 | RF01, RF07 |
| **Descrição** | O usuário recebe a solicitação de um passageiro para a sua carona cadastrada | | | |
| **Especificação** | **1 Fluxo Básico**  1.1 O usuário caroneiro recebe uma notificação quando algum usuário solicitou a carona cadastrada;  1.2 E visualiza os usuários ligados a carona cadastrada;  1.3 E retorna o pedido ao usuário solicitante.  **2 Fluxo Alternativo**  2.1 Retornar o pedido como negado. | | | |
|
|
|
|

Fonte: os autores (2017)

QUADRO 12 – REQUISITO FUNCIONAL X

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RF10 | Notificar retorno de pedido | P | 2 | RF01, RF09 |
| **Descrição** | O sistema envia uma notificação para o caroneiro notificando se foi aceito ou recusado. | | | |
| **Especificação** | **1 Fluxo Básico**  1.1 O usuário recebe um email, notificando-o sobre um passageiro que deseja a carona;  1.1 O usuário visualiza as informações do caronista.  **2 Fluxo Alternativo**  2.1 O usuário cancela pedido;  3.1 O usuário altera pedido. | | | |
|
|
|
|

Fonte: os autores (2017)

QUADRO 13 – REQUISITO FUNCIONAL XI

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF11 | Confiabilidade de informação | O | 2 | -- |
| **Descrição** | O sistema não poderá apresentar a outros usuários quaisquer informações consideradas confidenciais. | | | |

Fonte: os autores (2017)

QUADRO 14 - REQUISITO FUNCIONAL XII

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF12 | Reportar abuso | P | 2 | -- |
| **Descrição** | O usuário denuncia outro usuário com quem tenha feito uma viagem | | | |
| **Especificação** | **1 Fluxo Básico**  1.1 O usuário completa uma viagem;  1.2 Acessa a página de histórico de viagem clicando no menu esquerdo referente as suas viagens;  1.3 Encontra a viagem com o caroneiro ou caronista e denuncia;  **2 Fluxo Alternativo**  2.1 Feito a denúncia não é possível cancelar; | | | |

Fonte: os autores (2017)

QUADRO 15 - REQUISITO FUNCIONAL XIII

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF13 | Manter Minhas Caronas | P | 2 | -- |
| **Descrição** | O usuário (caronista) visualiza as caronas pendentes, aceitas e recusadas. | | | |
| **Especificação** | **1 Fluxo Básico**  1.1 O usuário clica no botão “Minhas caronas”;  1.2 O usuário acessa a lista de caronas finalizadas.  1.3 O usuário clica na tag “Pendentes” e é exibido as viagens pendentes.  1.4 O usuário clica na tag “Canceladas” e é exibido as viagens canceladas.  **2 Fluxo Alternativo**  2.1 | | | |

Fonte: os autores (2017)

QUADRO 16 - REQUISITO FUNCIONAL XIV

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF14 | Manter Minhas Viagens | P | 2 | -- |
| **Descrição** | O usuário (caroneiro) visualiza as caronas pendentes, aceitas e recusadas. | | | |
| **Especificação** | **1 Fluxo Básico**  1.1 O usuário clica no botão “Minhas viagens”;  1.2 O usuário acessa a lista de viagens finalizadas.  1.3 O usuário clica na tag “Pendentes” e é exibido as viagens pendentes.  1.4 O usuário clica na tag “Canceladas” e é exibido as viagens canceladas.  **2 Fluxo Alternativo**  2.1 | | | |

Fonte: os autores (2017)

QUADRO 17 - REQUISITO FUNCIONAL XV

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF15 | Manter Idioma | P | 2 | -- |
| **Descrição** | O usuário pode escolher o idioma de sua preferência. | | | |
| **Especificação** | **1 Fluxo Básico**  1.1 O usuário clica no botão de “Configurações”;  1.2 Escolhe o idioma que deseja ser exibido na tela;  **2 Fluxo Alternativo**  2.1 Não há; | | | |

Fonte: os autores (2017)

QUADRO 18 - REQUISITO FUNCIONAL XVI

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF16 | Manter Usuário | P | 2 | -- |
| **Descrição** | O usuário poderá editar seu apelido e e-mail de login. | | | |
| **Especificação** | **1 Fluxo Básico**  1.1 O usuário clica no botão “configurações”;  1.2 Clica na *tag* “Usuário”;  1.3 Edita suas informações;  **2 Fluxo Alternativo**  2.1 Retorno erro caso e-mail seja inválido ou igual a outro usuário; | | | |

Fonte: os autores (2017)

## REQUISITOS NÃO FUNCIONAIS

Condições como restrições, propriedades, segurança, comportamentos de todo ou parte do sistema, entram na lista dos requisitos não funcionais (RNF). Em outras palavras é aquilo que o sistema faz, e são considerados tão crítico e importante como os requisitos funcionais.

A seguir serão apresentados, do quadro 16 ao 21, os requisitos não funcionais do sistema.

QUADRO 19 - REQUISITO NÃO FUNCIONAL I

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF01 | Linguagem de desenvolvimento | P | 3 | -- |
| **Descrição** | O sistema será desenvolvido utilizando a linguagem de programação PHP, com auxílio do *framework* Laravel. | | | |

Fonte: os autores (2017)

QUADRO 20 - REQUISITO NÃO FUNCIONAL II

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF02 | Banco de Dados | P | 2 | -- |
| **Descrição** | O sistema deverá se comunicar com o Banco de Dados MySQL | | | |

Fonte: os autores (2017)

QUADRO 21 - REQUISITO NÃO FUNCIONAL III

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF03 | Internet | P | 2 | -- |
| **Descrição** | O sistema dependerá de conexão a internet | | | |

Fonte: os autores (2017)

QUADRO 22 - REQUISITO NÃO FUNCIONAL IV

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF04 | Sistema Responsivo | E | 1 | -- |
| **Descrição** | A interface deverá ser responsiva, adaptando-se a diversos dispositivos. | | | |

Fonte: os autores (2017)

QUADRO 23 - REQUISITO NÃO FUNCIONAL V

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF05 | Flexibilidade do sistema | E | 2 | -- |
| **Descrição** | O sistema deverá ser executado em qualquer navegador, dispositivo ou sistema operacional. | | | |

Fonte: os autores (2017)

QUADRO 24 - REQUISITO NÃO FUNCIONAL VI

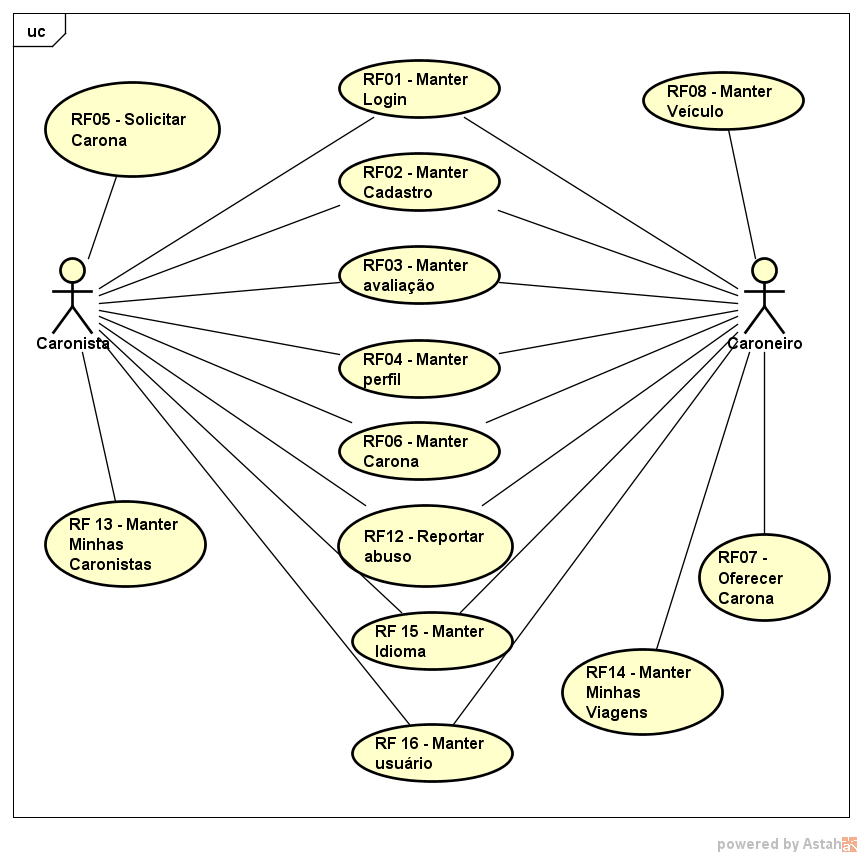
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF06 | Disponibilidade | O | 2 | -- |
| **Descrição** | O sistema deverá ter alta disponibilidade, ou seja, ficar 99% do tempo online. | | | |

Fonte: os autores (2017)

# CASO DE USO

Caso de uso, ou, *use* *case* é uma unidade da *Unified Modeling Language* (UML) que foca nas funcionalidades do sistema, a partir do ponto de vista do usuário. Segundo Jacobson (2007, p. 56), escritor, cientista da computação e mestre em engenharia eletrônica, um caso de uso é um “documento narrativo que descreve a sequência de eventos de um ator que usa um sistema para completar um processo”.

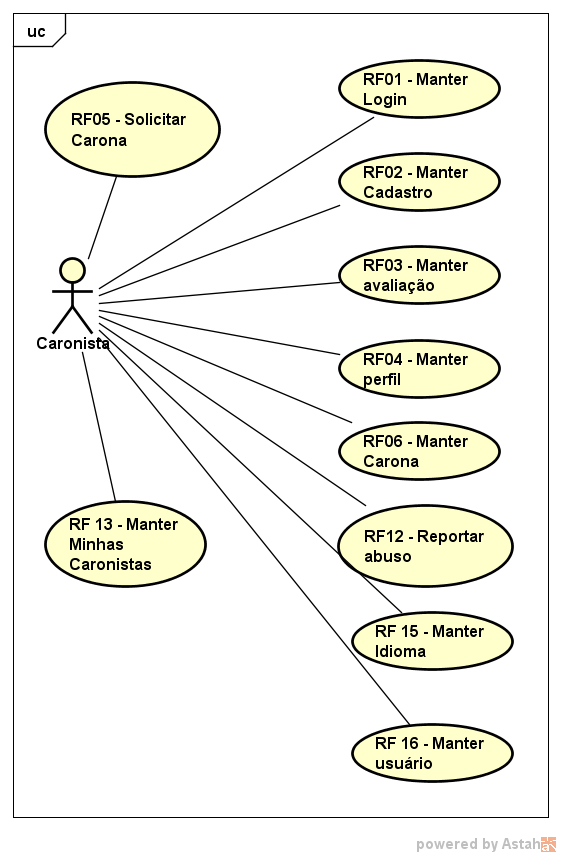
FIGURA 1 – DIAGRAMA GERAL DE CASO DE USO



Fonte: os autores (2017)

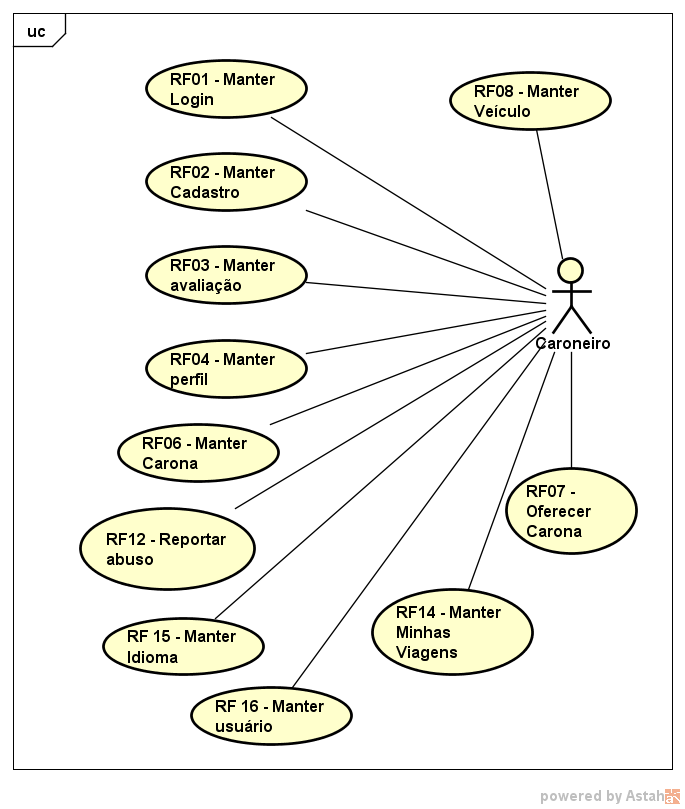
Os diagramas devem ser compreendidos por todos os *stakeholders –* esses são caracterizados por serem as partes interessadas no projeto. É no diagrama que, muitas vezes, o usuário terá uma compreensão maior do sistema que está adquirindo. Sendo, a seguir estão expostos os casos de uso de cada autor e seus relacionamentos.

FIGURA 2 – DIAGRAMA DE CASO DE USO CARONISTA



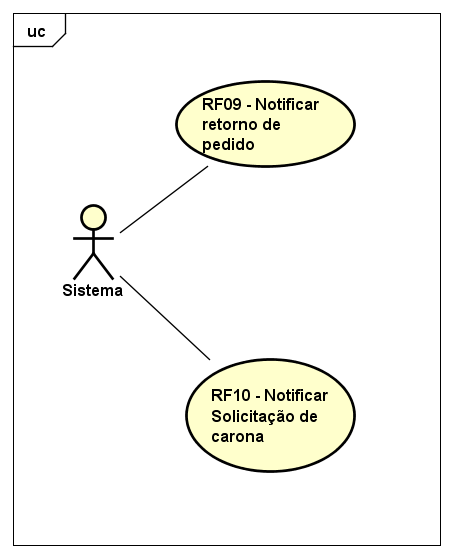
Fonte: os autores (2017)

FIGURA 3 – DIAGRAMA DE CASO DE USO CARONEIRO



Fonte: os autores (2017)

FIGURA 4 – DIAGRAMA DE CASO DE USO DO SISTEMA



Fonte: os autores (2017)

## ESPECIFICAÇÃO DE CASO DE USO

Quando o caso de uso geral não for suficiente para as partes interessadas entenderem por completo o *software*, as especificações de caso de uso são desenvolvidas com o intuito de garantir o entendimento completo do sistema. São as especificações para o sistema Bonjou, apresentadas nos quadros 22 a 30.

QUADRO 25 – ESPECIFICAÇÃO I

|  |  |
| --- | --- |
| **Nome:** | Manter Login |
| **Descrição:** | Permite que os usuários cadastrados tenham acesso ao sistema |
| **Atores:** | Usuário (caroneiro e caronista) |
| **Pré-Condições:** | O usuário deve possuir cadastro no sistema. |
| **Fluxo Básico:** | 1. O usuário acessa o site (www.bonjou.com.br)  2. O usuário seleciona o botão de Login  3. O sistema encaminha o usuário para a página de login.  4. O usuário informa login e senha  5. O usuário clica no botão "Entrar" para enviar seus dados.  6. O sistema verifica os dados digitados. |

Fonte: os autores (2017)

QUADRO 26 – ESPECIFICAÇÃO II

|  |  |
| --- | --- |
| **Nome:** | Manter Cadastro |
| **Descrição:** | Permite o usuário criar um cadastro e acessar as funcionalidades do sistema |
| **Atores:** | Usuário (caroneiro e caronista) |
| **Pré-Condições:** | Não possuir cadastro no sistema |
| **Fluxo Básico:** | 1. O usuário acessa o site (www.bonjou.com.br)  2. O usuário seleciona o botão de Cadastre-se.  3. Insere os seguintes dados: nome, email, senha e confirmar senha.  4. Clica no botão cadastrar. |

Fonte: os autores (2017)

QUADRO 27 – ESPECIFICAÇÃO III

|  |  |
| --- | --- |
| **Nome:** | Manter Avaliação |
| **Descrição:** | Permite que o caroneiro e imigrante avaliem as caronas realizadas. |
| **Atores:** | Usuário (caroneiro e caronista) |
| **Pré-Condições:** | Estar logado e ter finalizado uma carona. |
| **Fluxo Básico:** | 1. O usuário conclui a carona  2. Acessa os dados do histórico de caronas.  3. Acessa a carona que deseja avaliar.  4. Realiza uma avaliação do caroneiro/imigrante. |

Fonte: os autores (2017)

QUADRO 28 – ESPECIFICAÇÃO IV

|  |  |
| --- | --- |
| **Nome:** | Manter Perfil |
| **Descrição:** | Os usuários do sistema mantêm seus perfis atualizados. |
| **Atores:** | Usuário (caronista e caroneiro) |
| **Pré-Condições:** | Usuário estar autenticado no sistema |
| **Fluxo Básico:** | 1. Logado no sistema o usuário clica no botão “Meu perfil”.  2. E atualiza os dados como: Nome, telefone, foto, sobre, idade e gênero.  3. Clica no botão “Atualizar Perfil” |

Fonte: os autores (2017)

QUADRO 29 – ESPECIFICAÇÃO V

|  |  |
| --- | --- |
| **Nome:** | Manter Carona |
| **Descrição:** | O usuário verifica as caronas e solicita a carona desejada. |
| **Atores:** | Usuário |
| **Pré-Condições:** | Estar autenticado no sistema. |
| **Fluxo Básico:** | 1 Ator caroneiro clica no menu “Oferecer Caronas”.   * 1. O sistema exibe a tela de cadastro de caronas   2. O usuário preenche os campos com os seguintes dados: Data de partida, Hora de partida, local de partida, local de chegada, veículo e nº de passageiros;   3. Por fim, clica no botão Cadastrar.  1. Ator caronista clica no menu “Caronas”    1. A sistema apresenta uma lista de caronas disponíveis no momento.    2. O usuário clica sobre uma das viagens escolhidas.    3. O sistema apresenta dados detalhados sobre o motorista e a viagem.    4. O usuário clica no botão reservar carona.    5. O sistema notifica, via e-mail, o ator caronista, de que há uma solicitação de carona nova. |

Fonte: os autores (2017)

QUADRO 30 – ESPECIFICAÇÃO VI

|  |  |
| --- | --- |
| **Nome:** | Manter Veículo |
| **Descrição:** | Usuário cadastra, edita e exclui um veículo. |
| **Atores:** | Usuário (caroneiro) |
| **Pré-Condições:** | Estar autenticado no sistema. |
| **Fluxo Básico:** | 1. Entrar na página www.bonjou.com.br  2. Estar logado no sistema  3. Entrar na página “Meus veículos”  4. Clicar na opção representado pelo símbolo de “+”  5. Inserir os dados do carro que são: marca, modelo, ano, cor, placa e número de passageiros.  6. Clicar em “Salvar veículo”. |

Fonte: os autores (2017)

QUADRO 31 – ESPECIFICAÇÃO VIII

|  |  |
| --- | --- |
| **Nome:** | Notificar retorno de pedido |
| **Descrição:** | Sistema envia uma notificação para o motorista da viagem. |
| **Atores:** | Usuário (caroneiro). |
| **Pré-Condições:** | Estar logado e ser um motorista. |
| **Fluxo Básico:** | 1. Um usuário ingressa em uma carona existente  2. O sistema envia uma notificação para o motorista informando que tem mais um passageiro na carona |

Fonte: os autores (2017)

QUADRO 322 – ESPECIFICAÇÃO IX

|  |  |
| --- | --- |
| **Nome:** | Notificar solicitação de carona |
| **Descrição:** | Sistema envia uma notificação para o passageiro confirmando a carona. |
| **Atores:** | Usuário (caronista). |
| **Pré-Condições:** | Estar logado e ter solicitado carona. |
| **Fluxo Básico:** | 1. O motorista recebe a notificação do passageiro.  2. O sistema envia uma notificação para o passageiro informando se o seu pedido foi aceito ou recusado. |

Fonte: os autores (2017)

QUADRO 33 - ESPECIFICAÇÃO X

|  |  |
| --- | --- |
| **Nome:** | Visualizar histórico de viagens |
| **Descrição:** | Permite que o usuário veja as suas viagens realizadas |
| **Atores:** | Usuário (caroneiro e caronista). |
| **Pré-Condições:** | Estar logado e ter completado alguma viagem |
| **Fluxo Básico:** | 1. O usuário acessa a página de “Minhas viagens”  2. O sistema mostra as viagens em forma de lista de acordo com a data mais nova.  3. Ao clicar em uma das viagens o usuário terá mais detalhes da viagem e do usuário com quem fez a viagem selecionada. |

Fonte: os autores (2017)

QUADRO 34 - ESPECIFICAÇÃO XI

|  |  |
| --- | --- |
| **Nome:** | Denunciar abuso |
| **Descrição:** | Permite que o usuário denuncie um motorista ou passageiro |
| **Atores:** | Usuário (caroneiro e caronista). |
| **Pré-Condições:** | Estar logado e ter completado alguma viagem |
| **Fluxo Básico:** | 1. O usuário acessa a página de “Minhas viagens”  2. O sistema mostra as viagens em forma de lista de acordo com a data mais nova. |

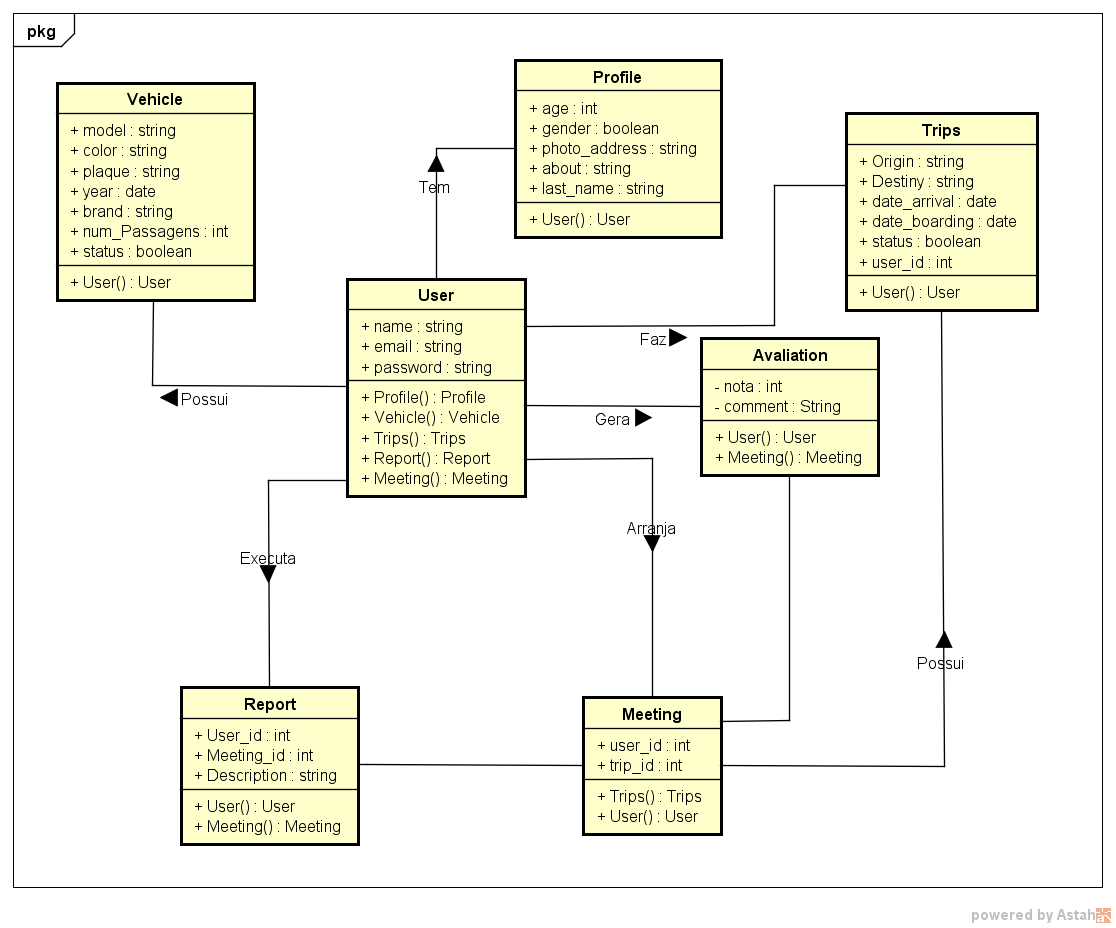
Fonte: os autores (2017)

# DIAGRAMA DE CLASSES

Em um diagrama de classes são mostradas as classes utilizadas no sistema com seus relacionamentos. Pode ser considerado o diagrama central da modelagem orientada a objetos.

A seguir é apresentado o diagrama utilizado no sistema junto a seu nome, atributos e métodos de cada classe, de acordo com as normas de UML, uma linguagem padrão para a elaboração de projetos de *softwares*. A figura 5 apresenta o diagrama de classes do sistema.

FIGURA 5 - DIAGRAMA DE CLASSE

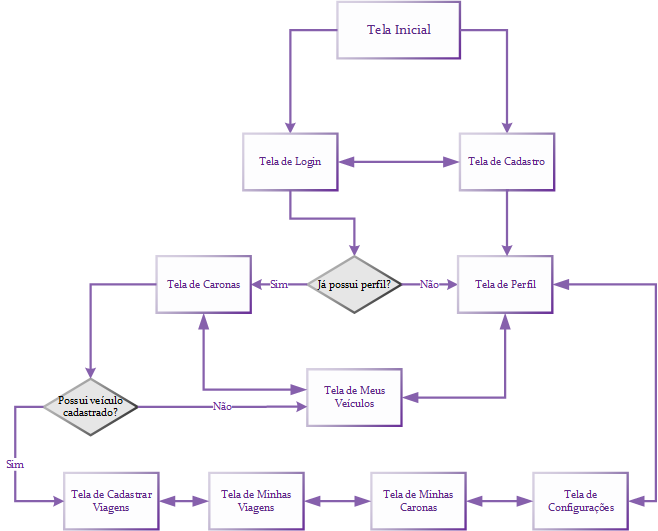


Fonte: os autores (2017)

# DIAGRAMA DE TELAS

Segue o fluxo das telas empregadas pelo sistema Bonjou.

FIGURA 6 - FLUXO DE TELAS

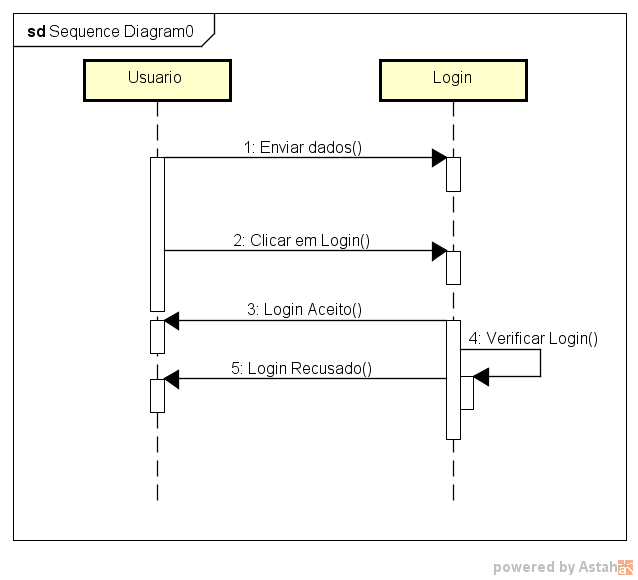


Fonte: os autores (2017)

# DIAGRAMA DE SEQUÊNCIA

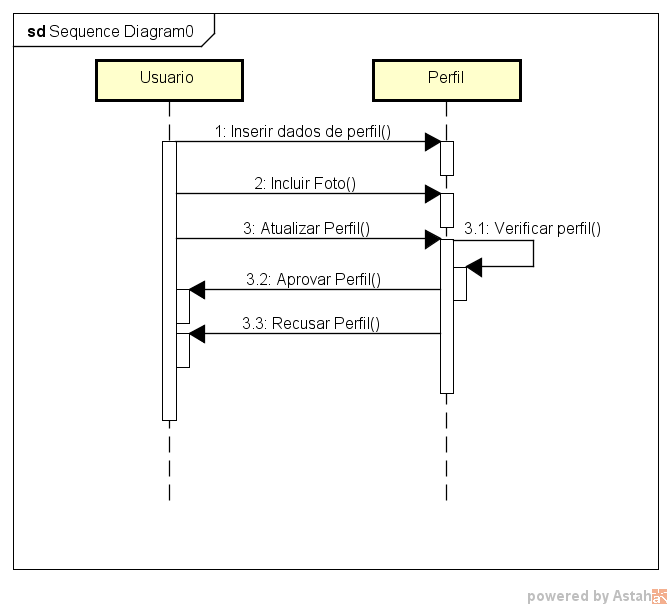
A partir da UML, linguagem padrão para projetos de *softwares*, um diagrama de sequência mostra a sequência de mensagens transmitidas entre os objetos. O diagrama compõe objetos, demonstrado por linhas de vida e as mensagens que são trocadas durante a interação.

FIGURA 7 - DIAGRAMA DE SEQUENCIA I



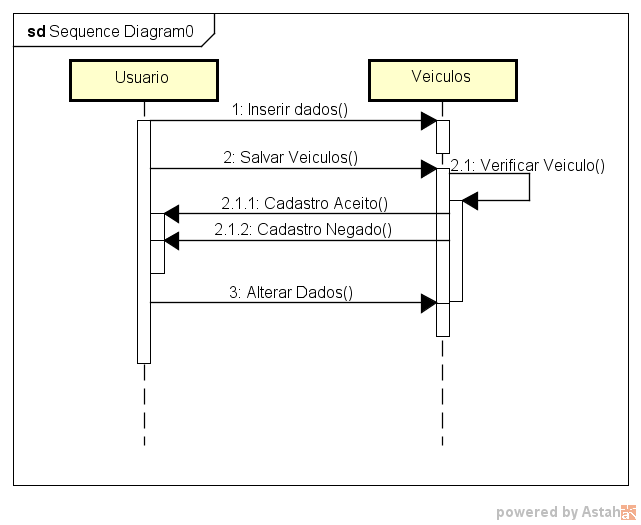
Fonte: os autores (2017)

FIGURA 8 - DIAGRAMA DE SEQUÊNCIA II



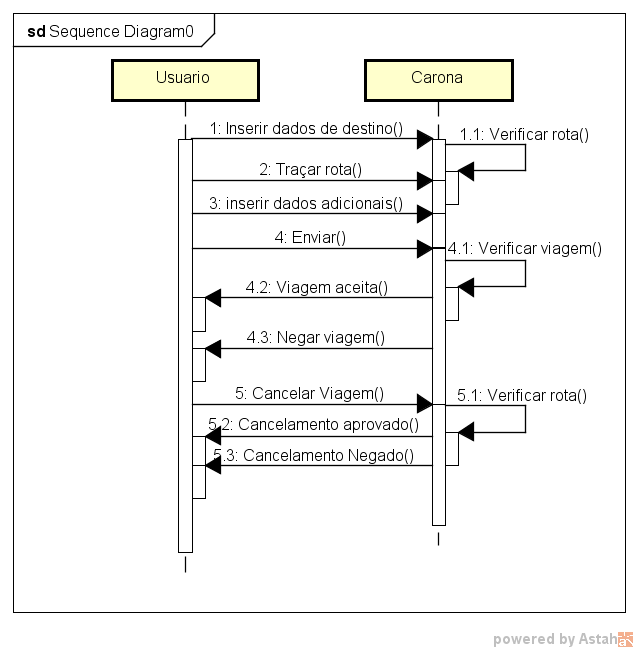
Fonte: os autores (2017)

FIGURA 9 - DIAGRAMA SEQUENCIA III



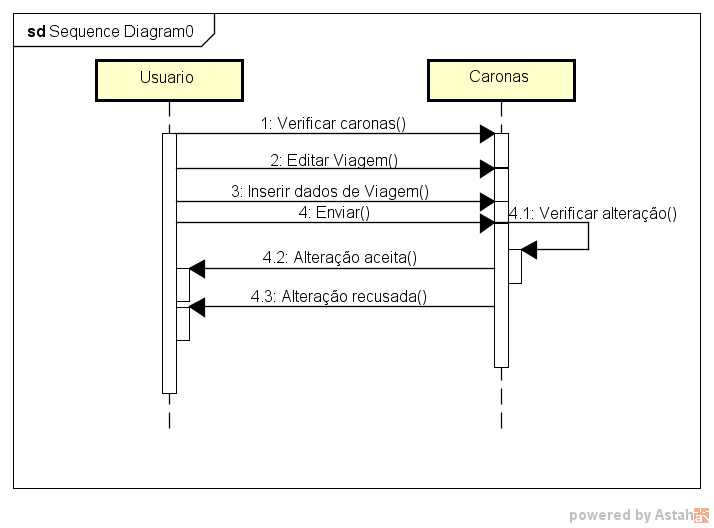
Fonte: os autores (2017)

FIGURA 10 - DIAGRAMA DE SEQUÊNCIA IV



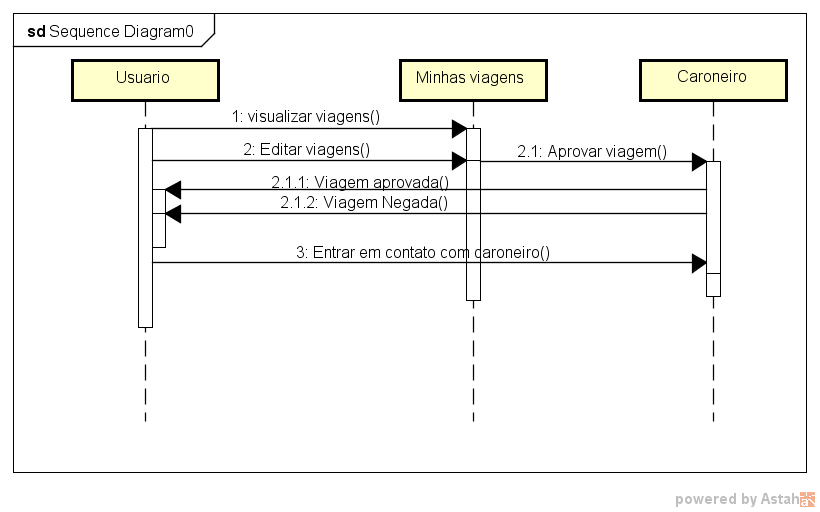
Fonte: os autores (2017)

FIGURA 11 - DIAGRAMA DE SEQUÊNCIA V



Fonte: os autores (2017)

FIGURA 12 - DIAGRAMA DE SEQUÊNCIA VI

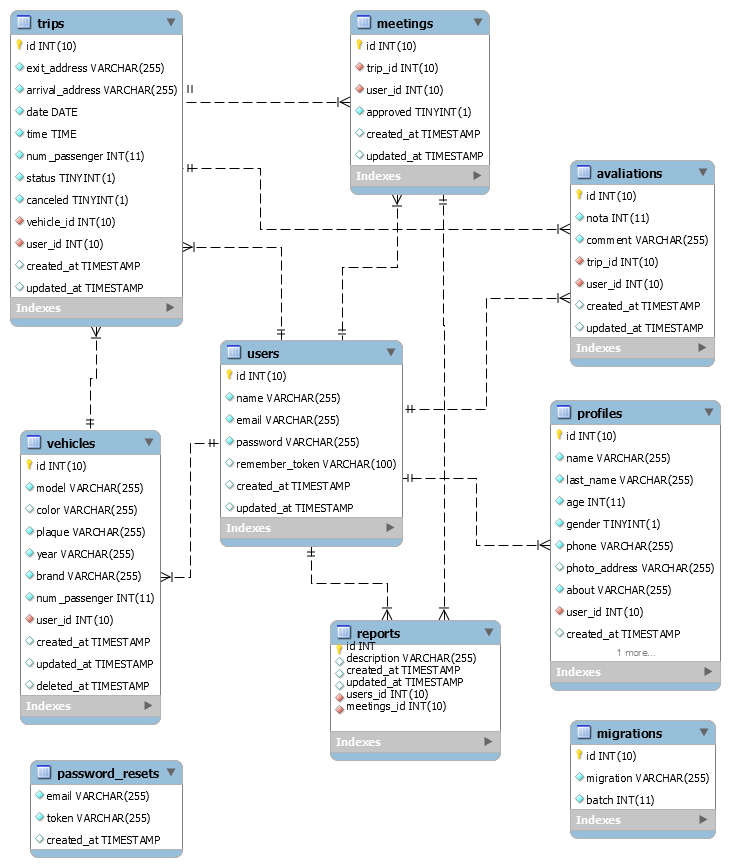


Fonte: os autores (2017)

# BANCO DE DADOS

As imagens a seguir representam o modelo do banco de dados que foi criado para o sistema. Suas interações e as cardinalidades dos relacionamentos.

FIGURA 13 – MODELO CONCEITUAL DO BANCO DE DADOS



Fonte: os autores (2017)

# SCRIPTS

De acordo com as tabelas mostradas no capítulo 7, esta seção indica apenas o script de criação de tabelas e demais comandos.

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

SET time\_zone = "+00:00";

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8mb4 \*/;

--

-- Database: `bonjou`

--

-- --------------------------------------------------------

--

-- Table structure for table `avaliations`

--

CREATE TABLE `avaliations` (

`id` int(10) UNSIGNED NOT NULL,

`nota` int(11) NOT NULL,

`comment` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`trip\_id` int(10) UNSIGNED NOT NULL,

`user\_id` int(10) UNSIGNED NOT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Table structure for table `meetings`

--

CREATE TABLE `meetings` (

`id` int(10) UNSIGNED NOT NULL,

`trip\_id` int(10) UNSIGNED NOT NULL,

`user\_id` int(10) UNSIGNED NOT NULL,

`approved` tinyint(1) NOT NULL DEFAULT '0',

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Table structure for table `migrations`

--

CREATE TABLE `migrations` (

`id` int(10) UNSIGNED NOT NULL,

`migration` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`batch` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `migrations`

--

INSERT INTO `migrations` (`id`, `migration`, `batch`) VALUES

(16, '2014\_10\_12\_000000\_create\_users\_table', 1),

(17, '2014\_10\_12\_100000\_create\_password\_resets\_table', 1),

(18, '2017\_09\_03\_182623\_create\_profiles\_table', 1),

(19, '2017\_09\_17\_181209\_create\_vehicles\_table', 1),

(20, '2017\_09\_17\_181455\_create\_trips\_table', 1),

(21, '2017\_09\_17\_183133\_create\_meetings\_table', 1),

(22, '2017\_10\_01\_184617\_create\_reports\_table', 1),

(23, '2017\_10\_13\_165916\_create\_avaliations\_table', 1);

-- --------------------------------------------------------

--

-- Table structure for table `password\_resets`

--

CREATE TABLE `password\_resets` (

`email` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`token` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`created\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Table structure for table `profiles`

--

CREATE TABLE `profiles` (

`id` int(10) UNSIGNED NOT NULL,

`name` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`last\_name` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`age` int(11) NOT NULL,

`gender` tinyint(1) NOT NULL,

`phone` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`photo\_address` varchar(255) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

`about` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`user\_id` int(10) UNSIGNED NOT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `profiles`

--

INSERT INTO `profiles` (`id`, `name`, `last\_name`, `age`, `gender`, `phone`, `photo\_address`, `about`, `user\_id`, `created\_at`, `updated\_at`) VALUES

(1, 'Matheus', 'Afornali', 23, 1, '(41) 99147-5805', '/img/resize/profile-1.jpg', 'Eu sou um cara legalzão.', 1, '2017-11-05 17:25:45', '2017-11-05 17:25:45');

-- --------------------------------------------------------

--

-- Table structure for table `reports`

--

CREATE TABLE `reports` (

`id` int(10) UNSIGNED NOT NULL,

`description` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`meeting\_id` int(10) UNSIGNED NOT NULL,

`user\_id` int(10) UNSIGNED NOT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Table structure for table `trips`

--

CREATE TABLE `trips` (

`id` int(10) UNSIGNED NOT NULL,

`exit\_address` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`arrival\_address` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`date` date NOT NULL,

`time` time NOT NULL,

`num\_passenger` int(11) NOT NULL,

`status` tinyint(1) NOT NULL DEFAULT '1',

`canceled` tinyint(1) NOT NULL DEFAULT '0',

`vehicle\_id` int(10) UNSIGNED NOT NULL,

`user\_id` int(10) UNSIGNED NOT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Table structure for table `users`

--

CREATE TABLE `users` (

`id` int(10) UNSIGNED NOT NULL,

`name` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`email` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`password` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`remember\_token` varchar(100) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `users`

--

INSERT INTO `users` (`id`, `name`, `email`, `password`, `remember\_token`, `created\_at`, `updated\_at`) VALUES

(1, 'Matheus Afornali', 'afornalimatheus@gmail.com', '$2y$10$jZFuKEqoLVXBXDsRYfmbP.Rfc5p99orV7iLSGsmw0AAE5LIYrlnKO', NULL, '2017-11-05 17:21:41', '2017-11-05 17:21:41');

-- --------------------------------------------------------

--

-- Table structure for table `vehicles`

--

CREATE TABLE `vehicles` (

`id` int(10) UNSIGNED NOT NULL,

`model` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`color` varchar(255) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

`plaque` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`year` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`brand` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`num\_passenger` int(11) NOT NULL,

`user\_id` int(10) UNSIGNED NOT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL,

`deleted\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

--

-- Dumping data for table `vehicles`

--

INSERT INTO `vehicles` (`id`, `model`, `color`, `plaque`, `year`, `brand`, `num\_passenger`, `user\_id`, `created\_at`, `updated\_at`, `deleted\_at`) VALUES

(1, '1010', 'Verde', 'ACL-7618', '1985', '23', 3, 1, '2017-11-05 17:27:10', '2017-11-05 17:27:10', NULL);

--

-- Indexes for dumped tables

--

--

-- Indexes for table `avaliations`

--

ALTER TABLE `avaliations`

ADD PRIMARY KEY (`id`),

ADD KEY `avaliations\_trip\_id\_foreign` (`trip\_id`),

ADD KEY `avaliations\_user\_id\_foreign` (`user\_id`);

--

-- Indexes for table `meetings`

--

ALTER TABLE `meetings`

ADD PRIMARY KEY (`id`),

ADD KEY `meetings\_trip\_id\_foreign` (`trip\_id`),

ADD KEY `meetings\_user\_id\_foreign` (`user\_id`);

--

-- Indexes for table `migrations`

--

ALTER TABLE `migrations`

ADD PRIMARY KEY (`id`);

--

-- Indexes for table `profiles`

--

ALTER TABLE `profiles`

ADD PRIMARY KEY (`id`),

ADD KEY `profiles\_user\_id\_foreign` (`user\_id`);

--

-- Indexes for table `reports`

--

ALTER TABLE `reports`

ADD PRIMARY KEY (`id`),

ADD KEY `reports\_meeting\_id\_foreign` (`meeting\_id`),

ADD KEY `reports\_user\_id\_foreign` (`user\_id`);

--

-- Indexes for table `trips`

--

ALTER TABLE `trips`

ADD PRIMARY KEY (`id`),

ADD KEY `trips\_vehicle\_id\_foreign` (`vehicle\_id`),

ADD KEY `trips\_user\_id\_foreign` (`user\_id`);

--

-- Indexes for table `users`

--

ALTER TABLE `users`

ADD PRIMARY KEY (`id`),

ADD KEY `users\_id\_index` (`id`);

--

-- Indexes for table `vehicles`

--

ALTER TABLE `vehicles`

ADD PRIMARY KEY (`id`),

ADD KEY `vehicles\_user\_id\_foreign` (`user\_id`);

--

-- AUTO\_INCREMENT for dumped tables

--

--

-- AUTO\_INCREMENT for table `avaliations`

--

ALTER TABLE `avaliations`

MODIFY `id` int(10) UNSIGNED NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT for table `meetings`

--

ALTER TABLE `meetings`

MODIFY `id` int(10) UNSIGNED NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT for table `migrations`

--

ALTER TABLE `migrations`

MODIFY `id` int(10) UNSIGNED NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=24;

--

-- AUTO\_INCREMENT for table `profiles`

--

ALTER TABLE `profiles`

MODIFY `id` int(10) UNSIGNED NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=2;

--

-- AUTO\_INCREMENT for table `reports`

--

ALTER TABLE `reports`

MODIFY `id` int(10) UNSIGNED NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT for table `trips`

--

ALTER TABLE `trips`

MODIFY `id` int(10) UNSIGNED NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT for table `users`

--

ALTER TABLE `users`

MODIFY `id` int(10) UNSIGNED NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=2;

--

-- AUTO\_INCREMENT for table `vehicles`

--

ALTER TABLE `vehicles`

MODIFY `id` int(10) UNSIGNED NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=2;

--

-- Constraints for dumped tables

--

--

-- Constraints for table `avaliations`

--

ALTER TABLE `avaliations`

ADD CONSTRAINT `avaliations\_trip\_id\_foreign` FOREIGN KEY (`trip\_id`) REFERENCES `trips` (`id`),

ADD CONSTRAINT `avaliations\_user\_id\_foreign` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`);

--

-- Constraints for table `meetings`

--

ALTER TABLE `meetings`

ADD CONSTRAINT `meetings\_trip\_id\_foreign` FOREIGN KEY (`trip\_id`) REFERENCES `trips` (`id`),

ADD CONSTRAINT `meetings\_user\_id\_foreign` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`);

--

-- Constraints for table `profiles`

--

ALTER TABLE `profiles`

ADD CONSTRAINT `profiles\_user\_id\_foreign` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`);

--

-- Constraints for table `reports`

--

ALTER TABLE `reports`

ADD CONSTRAINT `reports\_meeting\_id\_foreign` FOREIGN KEY (`meeting\_id`) REFERENCES `meetings` (`id`),

ADD CONSTRAINT `reports\_user\_id\_foreign` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`);

--

-- Constraints for table `trips`

--

ALTER TABLE `trips`

ADD CONSTRAINT `trips\_user\_id\_foreign` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`),

ADD CONSTRAINT `trips\_vehicle\_id\_foreign` FOREIGN KEY (`vehicle\_id`) REFERENCES `vehicles` (`id`);

--

-- Constraints for table `vehicles`

--

ALTER TABLE `vehicles`

ADD CONSTRAINT `vehicles\_user\_id\_foreign` FOREIGN KEY (`user\_id`) REFERENCES `users` (`id`);

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

# CONCLUSÃO

Ao longo do desenvolvimento deste trabalho, baseado nas discussões a respeito da imigração no mundo, bem como pesquisas realizadas pela ONU e a própria realidade percebida na cidade de Curitiba, resultou no planejamento e na implementação do sistema de *carpooling* Bonjou. Este, busca ser uma plataforma *online* de carona solidária, o qual permitirá o uso compartilhado de um automóvel particular por duas ou mais pessoas, com o objetivo de auxiliar o imigrante em sua adaptação ao novo meio no qual vive.

No momento, apenas o site encontra-se disponível para acesso, no entanto, a partir dos testes já realizados é possível depreender que o sistema não encontra obstáculos para exercer sua função. Acredita-se que o Bonjou será capaz de ser a ponte entre inúmeros cidadãos nativos e novos cidadãos. A plataforma é simples, intuitiva e será um excelente meio de comunicação para aqueles que querem ser solidários e para aqueles que necessitam de um pouco de solidariedade.

Por fim, sabe-se que muito ainda pode ser feito pelo sistema Bonjou, como, por exemplo, o desenvolvimento de novas funcionalidades, bem como aumento de segurança e desempenho. Ademais, a criação de um aplicativo para *smartphones* é bastante interessante, pois estabelece a conexão entre os usuários de modo mais eficiente. Entre os aprendizados, destacam-se as questões referentes ao trabalho em equipe, conhecimento de novas tecnologias, a possibilidade de desenvolver parcerias com diferentes setores da sociedade, bem como a oportunidade de aplicar os conhecimentos adquiridos em sala de aula ao longo do curso.

# REFERÊNCIAS

CYSNEIROS, Luiz Marcio. *Requisitos Não Funcionais:* da elicitação ao modelo conceitual. (Tese) – Doutorado em Ciências da Computação, da Pontifícia Universidade Católica (PUC). Disponível em: <http://www-di.inf.puc-rio.br/~julio/Tese%20-%205.pdf>. Acesso em: 16 set. 2017.

IBM Knowledge Center. *Diagramas de sequência.* Disponível em: <https://www.ibm.com/support/knowledgecenter/pt-br/SSCLKU\_7.5.5/com.ibm.xtools.sequence.doc/topics/cseqd\_v.html>. Acesso em: 20 set. 2017.

MEDEIROS, Higor*. Introdução ao padrão MVC.* Disponível em: <http://www.devmedia.com.br/introducao-ao-padrao-mvc/29308>. Acesso em: 16 set. 2017.

PHP. *O que é PHP?* Disponível em: <https://secure.php.net/manual/pt\_BR/intro-whatis.php>. Acesso em: 20 set. 2017.

UNITED NATIONS. *Sustainable Development.*244 million international migrants living abroad worldwide, new UN statistics reveal. January 12, 2016. Disponível em: <http://www.un.org/sustainabledevelopment/blog/2016/01/244-million-international-migrants-living-abroad-worldwide-new-un-statistics-reveal/>. Acesso em: 10 set. 2017.

UM GUIA DO CONHECIMENTO EM GERENCIAMENTO DE PROJETOS (GUIA PMBOK), São Paulo: Editora Saraiva, 2014. p.112

# APÊNDICE I – Relatório de Testes

**1. Introdução**

Este apêndice de plano de teste tem como objetivo documentar os resultados e informações para planejar e controlar os testes de verificação e validação do Sistema Bonjou. Os testes são para avaliar as qualidades do sistema e serão utilizadas para medir características como: desempenho, confiabilidade, funcionalidade, entre outras.

**2. Requisitos a serem testados**

Para realização de um dos planos de testes foram utilizados como referência os requisitos funcionais do sistema. Os seguintes requisitos foram testados: Manter login, Manter cadastro, Manter avaliação, Manter perfil, Solicitar carona, Manter Carona, Oferecer carona, Manter veículo, Notificar solicitação de carona, Notificar retorno de carona, Visualizar histórico de viagem e Reportar abuso.

A ordem os Casos de Teste (CT) estão na mesma ordem que apresentados acima. E estão apresentados em quadros com o número do caso de teste, objetivo, passos a serem realizados, critérios e por fim o resultado.

**3. Testes por requisitos**

|  |  |
| --- | --- |
| **Caso Nº** | CT01 – Realizar acesso ao Sistema |
| **Objetivo do Teste** | Verificar se o usuário consegue efetuar o Login. |
| **Passos** | 1. Acessar a página Inicial do Bonjou; 2. Clicar no botão de Login; 3. Ir para tela de login; 4. Informar o usuário e a senha; 5. Clicar no botão Login; |
| **Critérios de Êxito** | Verificar e-mail e senha se corresponde e logar no sistema, caso esteja errado pedir para tentar de novo. |
| **Resultados** | **05 Nov:** Corretos, tanto para senhas correspondentes e não correspondentes (informando o respectivo erro ao usuário) |
| **Caso Nº** | CT02 – Realizar um cadastro para o Sistema |
| **Objetivo do Teste** | Verificar se o usuário consegue realizar um cadastro. |
| **Passos** | 1. Acessar a página Inicial do Bonjou; 2. Clicar no botão de Login -> Registar; 3. Informar o Nome, Endereço de E-Mail, Senha e Confirmar Senha; 4. Clicar no botão Registrar |
| **Critérios de Êxito** | Verificar se as senhas estão batendo, se o e-mail está válido e redirecionar para a página inicial do Bonjou. |
| **Resultados** | **05 Nov:** Correto, inclusive para verificação senha correspondente para autenticação, e redirecionamento para a tela principal. |

|  |  |
| --- | --- |
| **Caso Nº** | CT03 – Realizar uma avaliação de outros usuários |
| **Objetivo do Teste** | Verificar se o usuário consegue avaliar um usuário com quem tenha feito uma viagem. |
| **Passos** | 1. Acessar de minhas viagens; 2. Clicar com o botão esquerdo do mouse sobre uma viagem; 3. Colocar a quantidade de estrelas: 1 – Muito ruim até 5 – Muito bom, clicando em cima das estrelas; |
| **Critérios de Êxito** | Verificar se foi cadastrada com sucesso a nota da avaliação. |
| **Resultados** | **05 Nov:** Inconsistente, não há botão para avaliar usuário.  **15 Nov**: Inconsistente, não há botão para avaliar usuário. |

|  |  |
| --- | --- |
| **Caso Nº** | CT04 – Atualizar Perfil do usuário |
| **Objetivo do Teste** | Verificar se o usuário consegue alterar informações de seu perfil. |
| **Passos** | 1. Clicar no ícone referente a uma pessoa no canto superior direito; 2. Clicar em Meu Perfil; 3. Colocar o número correspondente da nota para a pessoa; 4. Adicionar as informações. |
| **Critérios de Êxito** | Verificar se foi atualizado as informações, verificar se a foto foi adicionada e alterada, não poder salvar perfil sem Nome. |
| **Resultados** | **05 Nov**: Correto, com verificação de campos obrigatórios. |

|  |  |
| --- | --- |
| **Caso Nº** | CT05 – Caronista solicitar uma viagem |
| **Objetivo do Teste** | Verificar se o usuário consegue solicitar a carona escolhida. |
| **Passos** | 1. Clicar em “Procurar Caronas”; 2. Clicar em uma das viagens listadas; 3. Clicar no botão “Reservar viagem”; |
| **Critérios de Êxito** | O caronista não poderá pedir uma viagem já finalizada, ou sem vagas. |
| **Resultados** | **05 Nov**: Inconsistente, aceita viagens sem nenhuma vaga disponível.  **15 Nov**: Correto, quando não possui mais vagas há um retorno do sistema avisando que não há uma vaga mais para aquela viagem. |

|  |  |
| --- | --- |
| **Caso Nº** | CT06 – Usuário cadastrar um veículo |
| **Objetivo do Teste** | Verificar se o usuário consegue cadastrar e editar um veículo. |
| **Passos** | 1. Entrar na página de veículos através do botão “Meus veículos”; 2. Clicar no ícone de “+” no canto superior direito; 3. Adicionar as informações do veículo. 4. Clicar em Salvar veículo; |
| **Critérios de Êxito** | Verificar se é possível adicionar um veículo sem nenhuma informação, se é possível editar as informações depois de salvas ou excluir veículo. |
| **Resultados** | **05 Nov:** Inconsistente, grava os dados com sucesso, porém aceita números no campo cor.  **15 Nov:** Correto. |

|  |  |
| --- | --- |
| **Caso Nº** | CT07 – Notificação de carona |
| **Objetivo do Teste** | Verificar se o e-mail de pedido de um usuário foi enviado para o dono da viagem. |
| **Passos** | 1. Entrar na página de viagens “Procurar Caronas”; 2. Clicar sobre a viagem desejada; 3. Verificar as informações e clicar no botão “Reservar Viagem” |
| **Critérios de Êxito** | O dono que cadastrou a viagem deverá receber um e-mail com as informações de quem requisitou uma carona, juntamente com um link para fácil acesso. |
| **Resultados** | **05 Nov:** Correto e com notificação diretamente no aplicativo, no campo caronas. |

|  |  |
| --- | --- |
| **Caso Nº** | CT08 – Notificar feedback da carona |
| **Objetivo do Teste** | Verificar se o e-mail de retorno da carona foi enviado. |
| **Passos** | 1. Dono da carona entra na página atraves do link do e-mail recebido; 2. Verifica os dados do caronista; 3. Clica no botão de aceitar ou recusar; |
| **Critérios de Êxito** | O caronista deverá receber um e-mail avisando-o se seu pedido de carona foi aceito ou recusado pelo dono da viagem. |
| **Resultados** | **05 Nov:** Correto, sem pendências. |

|  |  |
| --- | --- |
| **Caso Nº** | CT09 – Visualizar histórico de viagens realizadas |
| **Objetivo do Teste** | Verificar se o usuário consegue visualizar as viagens que foram completadas por ele. |
| **Passos** | 1. Entrar na página Minhas viagens; |
| **Critérios de Êxito** | O usuário deverá ter uma lista com todas as caronas pegas por ele, incluindo os detalhes de cada uma (ao clicar sobre uma delas). |
| **Resultados** | **05 Nov:** Correto, inclusive com as viagens geradas como caroneiro e caronista |

|  |  |
| --- | --- |
| **Caso Nº** | CT10 – Reportar abuso de um integrante da viagem |
| **Objetivo do Teste** | Verificar se o usuário consegue reportar algum abuso de um usuário incluindo um comentário. |
| **Passos** | 1. Acessar a página de “Minhas Viagens”; 2. Clicar sobre uma das viagens; 3. Clicar no botão “Reportar abuso” |
| **Critérios de Êxito** | O usuário poderá, ao concluir a viagem, reportar um abuso contra um integrante. |
| **Resultados** | **05 Nov:** Inconsistente, campo não inserido.  **15 Nov**: Inconsistente, campo não inserido. |

|  |  |
| --- | --- |
| **Caso Nº** | CT10 – Tradução de páginas |
| **Objetivo do Teste** | Verificar se o usuário consegue alterar o idioma das páginas. |
| **Passos** | 1. Acessar página de configurações do sistema; 2. Clicar no botão correspondendo ao idioma que deseja; |
| **Critérios de Êxito** | O sistema deverá trocar automaticamente para o idioma desejado pelo usuário. |
| **Resultados** | **05 Nov:** Opção de alterar idioma não inserido.  **15 Nov:** Correto, o sistema troca de idioma e mantém o mesmo. |

**4. Teste de Usabilidade**

O teste de usabilidade tem o objetivo de analisar a viabilidade e facilidade da manipulação do sistema, pelo ponto de vista do usuário. Para este teste foram utilizados os seguintes critérios de avaliação.

Menu – Símbolos diferentes dos tradicionais, navegabilidade, confusão de apresentação.

Sistema lento – Gerar informações e atualização de página se demoraram para carregar.

Página Inicial – Se a quantidade de informação sobre o aplicativo, ou se é possível entender a proposta do sistema.

Disposição de botões – Se os botões estão localizados em áreas intuitivas para pessoas que nunca utilizaram o aplicativo.

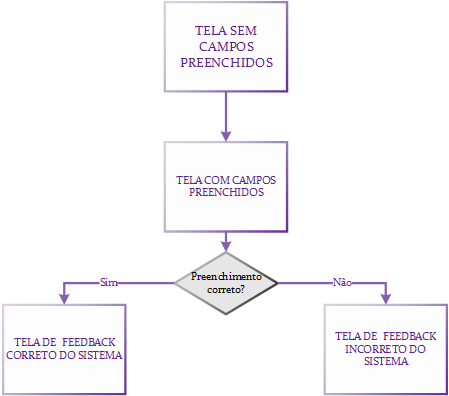
Sistema legível – Verifica se as cores da fonte, plano de fundo, imagens possuem contraste, tamanho adequado ou ficam distorcidas em aparelhos diferentes.

Para verificar esses critérios foram realizados testes no dia 05 e 15 de novembro com pessoas que não fizeram parte do desenvolvimento do aplicativo e que possuem pouca a média familiaridade com aplicativos.

# APÊNDICE II – Telas

Neste apêndice será apresentado o feedback do sistema quando há respostas corretas do usuário e também quando ocorre algum erro, seja feito pelo usuário ou algo inesperado do sistema.

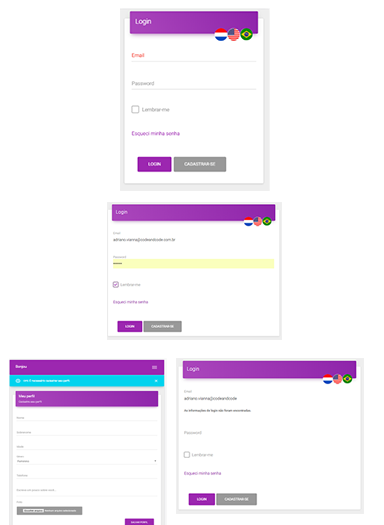
Para isso, as imagens a seguir consistirão em: A primeira imagem com a tela “vazia”, isto é, sem nenhum dos campos preenchidos em seguida outra imagem com campos preenchidos corretamente e o seu resultado, do outro lado da imagem dos campos corretas, uma imagem com os campos preenchidos errados e posteriormente seu erro. Para melhor entendimento segue o exemplo abaixo.



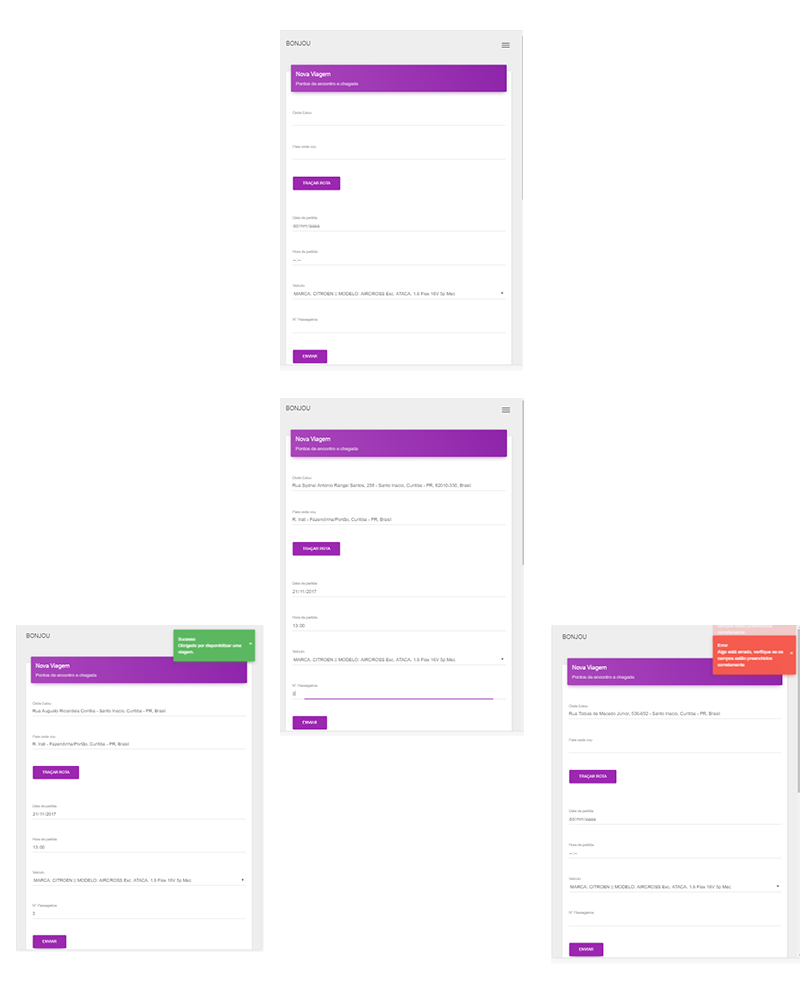
Fonte: os autores (2017)

A seguir será mostrada a tela de login e de viagem do sistema, para as outras telas não divulgadas a maneira que o sistema se comporta com a ação do usuário são as mesmas. Ou seja, ou o sistema aponta o erro para o usuário ou o mesmo nem permitirá que faça um erro, medida essa de prevenção.

Mapeamento de Tela de Login



Fonte: os autores (2017)

Mapeamento de Tela de Viagem

Fonte: os autores (2017)

1. No dialeto crioulo, utilizado pelos haitianos, a palavra *Bonjou*, derivada do francês *Bonjour*, a qual significa ‘Bom dia’, foi adotada para este trabalho em virtude do objetivo do sistema de aproximar imigrantes e moradores nativos. [↑](#footnote-ref-1)