**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 15](#_Toc497040138)

[FIGURA 2 – DIAGRAMA DE CASO DE USO CARONISTA 16](#_Toc497040139)

[FIGURA 3 – DIAGRAMA DE CASO DE USO CARONEIRO 17](#_Toc497040140)

[FIGURA 4 – DIAGRAMA DE CASO DE USO DO SISTEMA 17](#_Toc497040141)

[FIGURA 5 - DIAGRAMA DE CLASSE 20](#_Toc497040142)

[FIGURA 6 – MODELO CONCEITUAL DO BANCO DE DADOS 23](#_Toc497040143)

**LISTA DE QUADROS**

[Quadro 1 – Tipo do requisito 8](#_Toc497066928)

[Quadro 2 – Prioridade do requisito 8](#_Toc497066929)

[QUADRO 3 – REQUISITO FUNCIONAL I 9](#_Toc497066930)

[QUADRO 4 – REQUISITO FUNCIONAL II 9](#_Toc497066931)

[QUADRO 5 – REQUISITO FUNCIONAL III 9](#_Toc497066932)

[QUADRO 6 – REQUISITO FUNCIONAL IV 10](#_Toc497066933)

[QUADRO 7 – REQUISITO FUNCIONAL V 10](#_Toc497066934)

[QUADRO 8 – REQUISITO FUNCIONAL VI 10](#_Toc497066935)

[QUADRO 9 – REQUISITO FUNCIONAL VII 11](#_Toc497066936)

[QUADRO 10 – REQUISITO FUNCIONAL XIII 11](#_Toc497066937)

[QUADRO 11 – REQUISITO FUNCIONAL XIX 12](#_Toc497066938)

[QUADRO 12 – REQUISITO FUNCIONAL X 12](#_Toc497066939)

[QUADRO 13 – REQUISITO FUNCIONAL XI 12](#_Toc497066940)

[QUADRO 14 – REQUISITO FUNCIONAL XII 12](#_Toc497066941)

[QUADRO 15 – REQUISITO FUNCIONAL XIII 13](#_Toc497066942)

[QUADRO 16 – REQUISITO NÃO FUNCIONAL I 13](#_Toc497066943)

[QUADRO 17 – REQUISITO NÃO FUNCIONAL II 14](#_Toc497066944)

[QUADRO 18 – REQUISITO NÃO FUNCIONAL III 14](#_Toc497066945)

[QUADRO 19 – REQUISITO NÃO FUNCIONAL IV 14](#_Toc497066946)

[QUADRO 20 – REQUISITO NÃO FUNCIONAL V 14](#_Toc497066947)

[QUADRO 21 – REQUISITO NÃO FUNCIONAL VI 14](#_Toc497066948)

[QUADRO 22 – ESPECIFICAÇÃO I 18](#_Toc497066949)

[QUADRO 23 – ESPECIFICAÇÃO II 18](#_Toc497066950)

[QUADRO 24 – ESPECIFICAÇÃO III 18](#_Toc497066951)

[QUADRO 25 – ESPECIFICAÇÃO IV 19](#_Toc497066952)

[QUADRO 26 – ESPECIFICAÇÃO V 19](#_Toc497066953)

[QUADRO 27 – ESPECIFICAÇÃO VI 19](#_Toc497066954)

[QUADRO 28 – ESPECIFICAÇÃO VII 19](#_Toc497066955)

[QUADRO 29 – ESPECIFICAÇÃO VIII 20](#_Toc497066956)

[QUADRO 30 – ESPECIFICAÇÃO IX 20](#_Toc497066957)

[QUADRO 31 – ESPECIFICAÇÃO X 20](#_Toc497066958)

[QUADRO 32 – ESPECIFICAÇÃO XI 20](#_Toc497066959)

**SUMÁRIO**

[1 INTRODUÇÃO 6](#_Toc496287246)

[2 VISÃO GERAL DO SISTEMA 7](#_Toc496287247)

[2.1 FERRAMENTAS ADICIONAIS 7](#_Toc496287248)

[3 REQUISITOS DO SISTEMA 8](#_Toc496287249)

[3.1 REQUISITOS FUNCIONAIS 8](#_Toc496287250)

[3.2 REQUISITOS NÃO FUNCIONAIS 13](#_Toc496287251)

[4 CASO DE USO 15](#_Toc496287252)

[4.1 ESPECIFICAÇÃO DE CASO DE USO 18](#_Toc496287253)

[5 DIAGRAMA DE CLASSES 21](#_Toc496287254)

[6 DIAGRAMA DE SEQUÊNCIA 22](#_Toc496287255)

[7 BANCO DE DADOS 23](#_Toc496287256)

[8 SCRIPTS 24](#_Toc496287257)

[9 CONCLUSÃO 32](#_Toc496287258)

[REFERÊNCIAS 33](#_Toc496287259)

# 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 | 1 | RF02 |
| **Descrição** | O usuário ou administrador fazem *login* no site | | | |
| **Especificação** | 1. **Fluxo Básico:**    1. O usuário ou administrador acessam a *url* do sistema;    2. O usuário ou administrador seleciona o botão login;    3. O usuário ou administrador preenchem os campos com seus e-mail e senha;    4. O sistema verifica os dados, confere se existem e concede as devidas permissões; 2. **Fluxo Alternativo:**    1. A tela de *login* possui o botão – Registro;    2. A tela de *login* possui o link – 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;    2. O usuário entra com seus dados (nome, e-mail, senha, confirmar senha);    3. O usuário confirma se os dados foram adicionados corretamente;    4. Clica no botão Registrar. 2. **Fluxo Alternativo:**   1.1 O usuário altera o nome na tela de Perfil. | | | |

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 | 1 | RF01 |
| **Descrição** | O usuário mantém seu cadastro atualizado. | | | |
| **Especificação** | **1 Fluxo Básico**   * 1. O usuário clica no botão “Meu perfil”   2. E atualiza os seguintes dados:   - Nome  - Sobrenome  - Idade  - Gênero  - Sobre mim  - Telefone  - Documento de Identificação  - País  - Estado  - Cidade  - Foto (Imagem)  **2 Fluxo Alternativo**  2.1 O usuário clica no botão “Excluir perfil”;  2.2 O usuário clica no botão “Alterar perfil”. | | | |

Fonte: os autores (2017)

QUADRO 7 – REQUISITO FUNCIONAL V

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RF05 | Solicitar Carona | P | 1 | RF01 |
| **Descrição** | O usuário solicita carona | | | |
| **Especificação** | **1 Fluxo Básico**   * 1. O usuário clica no botão “Procurar carona”;   2. Seleciona a localização de destino;   3. Usuário localiza as caronas disponíveis;   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 agendamento | 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 | RF09 |
| **Descrição** | O usuário cadastra uma oferta de carona | | | |
| **Especificação** | **1 Fluxo Básico**   * 1. O usuário clica no botão “Oferecer carona”;   2. O usuário cadastra o local de saída;   3. O usuário cadastra o local de destino;   4. O usuário cadastra a data da carona;   5. O usuário cadastra o horário de partida.   **2 Fluxo Alternativo**  2.1 O usuário cancela a oferta de carona;  2.2 O usuário altera a oferta de carona.  **3 Fluxo Exceção**  3.1 Se o usuário não tem um veículo cadastrado, ele será direcionado a tela de “cadastrar veículo”; | | | |

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 Cadastrar veículo;   2. 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 | 1 | RF01, RF09 |
| **Descrição** | O usuário recebe a solicitação de carona 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 | 1 | 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 uma notificação sobre o status da solicitação de carona;  1.1 O usuário visualiza todas as solicitações abertas com retornos.  **2 Fluxo Alternativo**  2.1 O usuário cancela pedido;  2.2 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 | Visualizar histórico de viagens | P | 1 | -- |
| **Descrição** | **1 Fluxo Básico**  1.1 O usuário clica no botão “Minhas Viagens”;  1.1 O usuário visualiza todas as viagens realizadas por ele.  1.3 Ao clicar em cima de uma das viagens o usuário pode ver todas as informações referentes a viagem e do usuário com quem realizou ela;  **2 Fluxo Alternativo**  2.1 O Não há; | | | |

Fonte: os autores (2017)

QUADRO 15 - REQUISITO FUNCIONAL XIII

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF13 | 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 denuncia não é possível cancelar; | | | |

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 16 - REQUISITO NÃO FUNCIONAL I

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF01 | Linguagem de desenvolvimento | P | 1 | -- |
| **Descrição** | O sistema será desenvolvido utilizando a linguagem de programação PHP, com auxílio do *framework* Laravel. | | | |

Fonte: os autores (2017)

QUADRO 17 - REQUISITO NÃO FUNCIONAL II

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF02 | Banco de Dados | P | 1 | -- |
| **Descrição** | O sistema deverá se comunicar com o Banco de Dados MySQL | | | |

Fonte: os autores (2017)

QUADRO 18 - 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 19 - 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 20 - 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 21 - REQUISITO NÃO FUNCIONAL VI

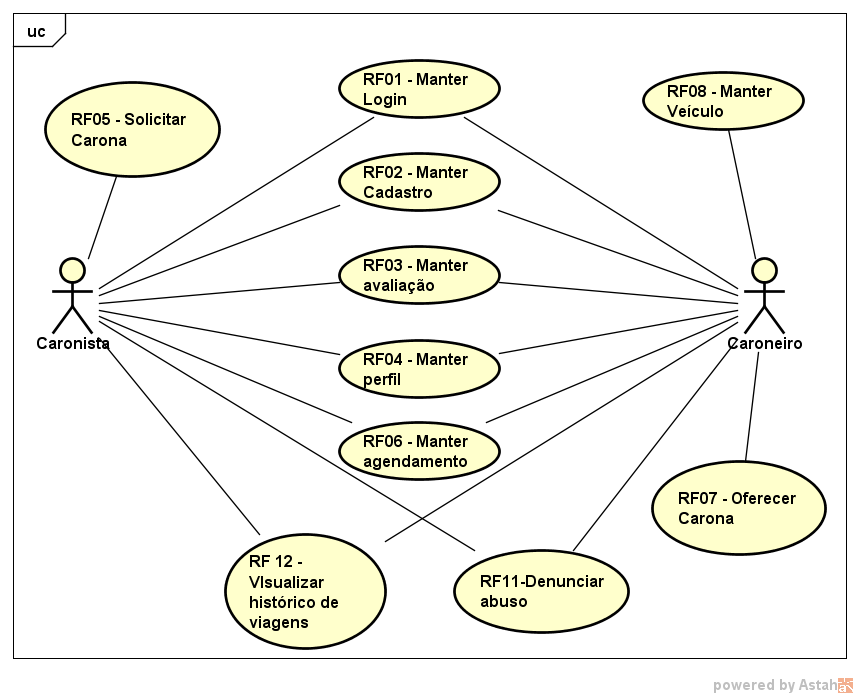
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Identificador** | **Nome** | **Tipo** | **Prioridade** | **Depende de** |
| RNF06 | Disponibilidade | O | 2 | -- |
| **Descrição** | O sistema deverá ter alta disponibilidade, ou seja, ficar 99% do tempo online. | | | |

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”.

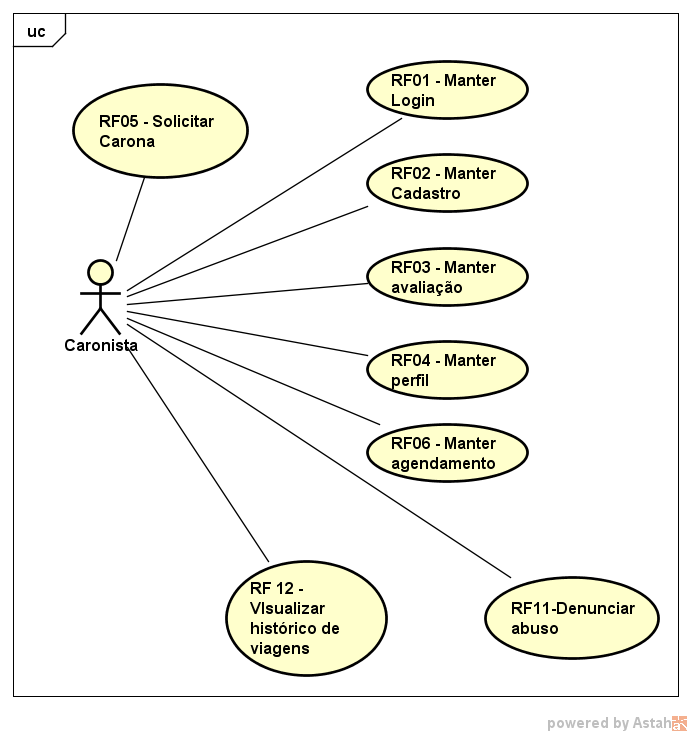
FIGURA 1 – DIAGRAMA GERAL DE CASO DE USO



Fonte: os autores (2017)

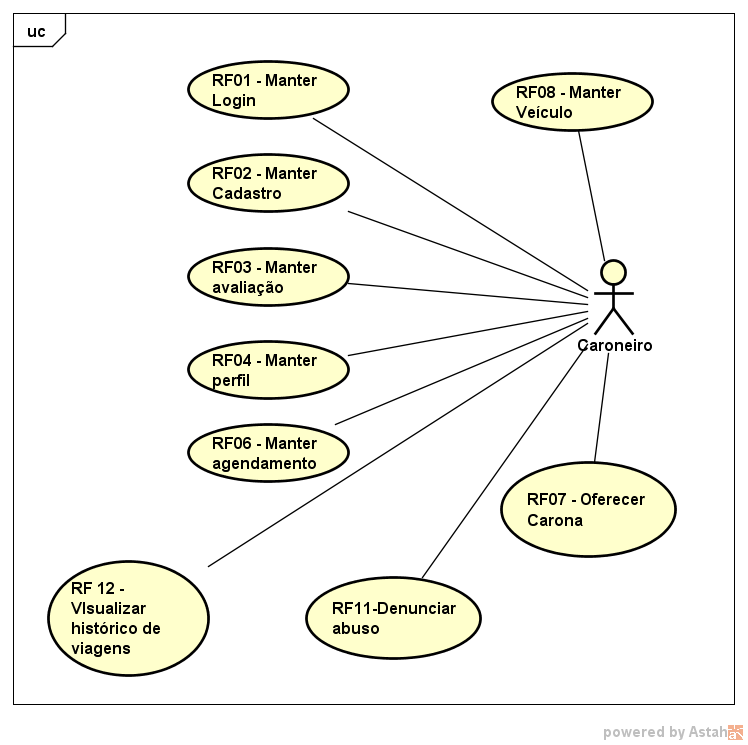
Os diagramas devem ser compreendidos por todos os *stakeholders –* esses são caracterizados por serem as partes interessadas no projeto. É no diagrama que, muitas vezes, o usuário terá uma compreensão maior do sistema que está adquirindo. Sendo, a seguir estão expostos os casos de uso de cada autor e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

FIGURA 2 – DIAGRAMA DE CASO DE USO CARONISTA



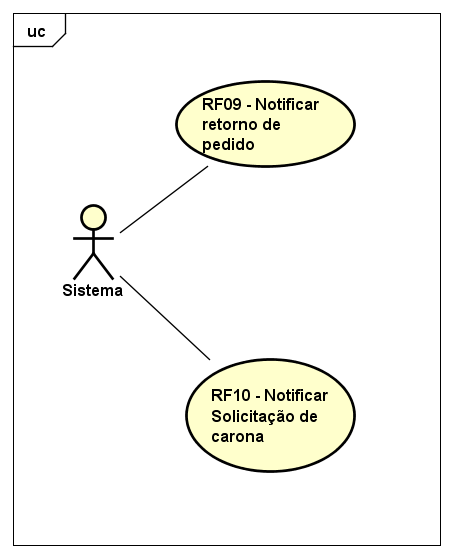
Fonte: os autores (2017)

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 22 – ESPECIFICAÇÃO I

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

Fonte: os autores (2017)

QUADRO 23 – 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 Registro.  3. Insere os seguintes dados: nome, email e senha.  4. Clica no botão cadastrar. |

Fonte: os autores (2017)

QUADRO 24 – 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 25 – 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. |

Fonte: os autores (2017)

QUADRO 26 – ESPECIFICAÇÃO V

|  |  |
| --- | --- |
| **Nome:** | Solicitar Carona |
| **Descrição:** | O usuário verifica as caronas e solicita a carona desejada. |
| **Atores:** | Usuário (coranista). |
| **Pré-Condições:** | Estar autenticado no sistema e não possuir veículo cadastrado. |
| **Fluxo Básico:** | 1. Usuário (imigrante) acessa a página com as caronas oferecidas.  2. Clica em “Solicitar carona” na opção que melhor se adapta. |

Fonte: os autores (2017)

QUADRO 27 – ESPECIFICAÇÃO VI

|  |  |
| --- | --- |
| **Nome:** | Manter Veículo |
| **Descrição:** | Usuário cadastra, edita e exclui um veículo. |
| **Atores:** | Usuário (caroneiro) |
| **Pré-Condições:** | Estar logado 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 “Cadastrar um veículo”  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 28 – ESPECIFICAÇÃO VII

|  |  |
| --- | --- |
| **Nome:** | Manter Agendamento |
| **Descrição:** | Usuário pode realizar um novo agendamento, editar um já existente e também excluir um agendamento. |
| **Atores:** | Usuário |
| **Pré-Condições:** | Estar logado no sistema. |
| **Fluxo Básico:** | 1. Usuário tem acesso a lista de viagens existentes  2. Seleciona a viagem desejada  3. O sistema realiza uma verificação para identificar se há vagas disponíveis  4. O sistema envia uma notificação para o motorista informando que mais um passageiro ingressou na sua viagem. |

Fonte: os autores (2017)

QUADRO 29 – 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 30 – 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 31 - 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 pagina 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 32 - 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 pagina de “Minhas viagens”  2. O sistema mostra as viagens em forma de lista de acordo com a data mais nova. |

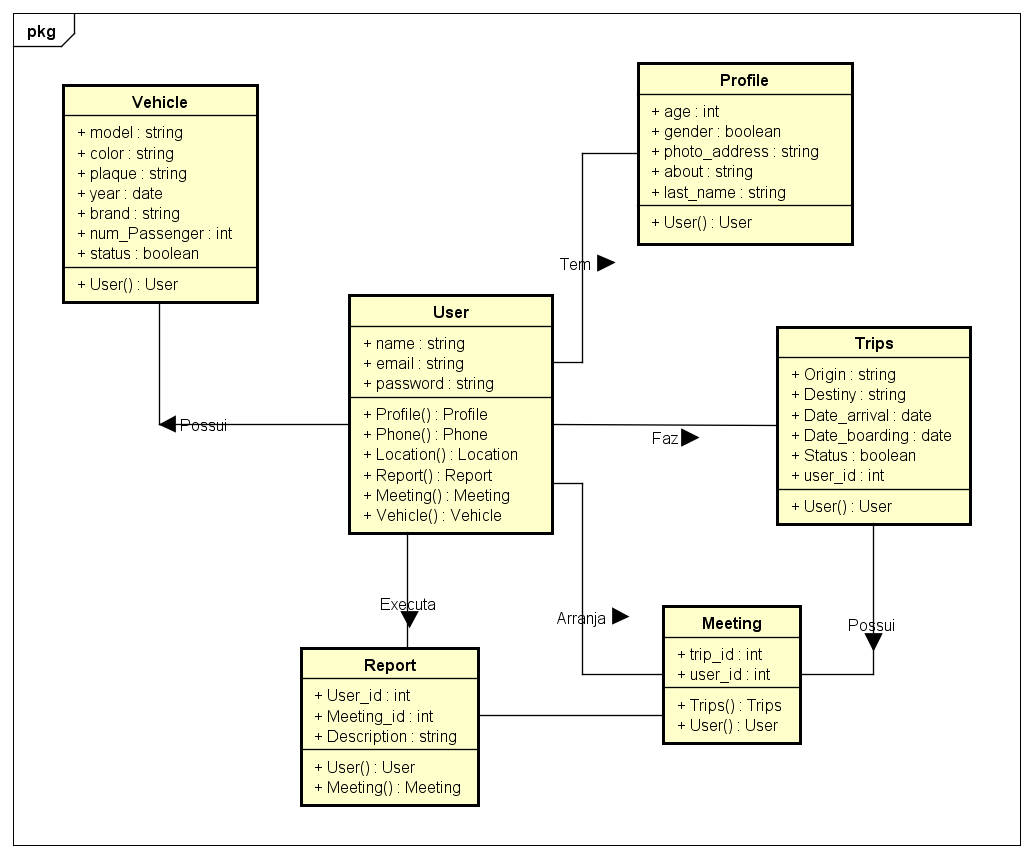
Fonte: os autores (2017)

# 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)

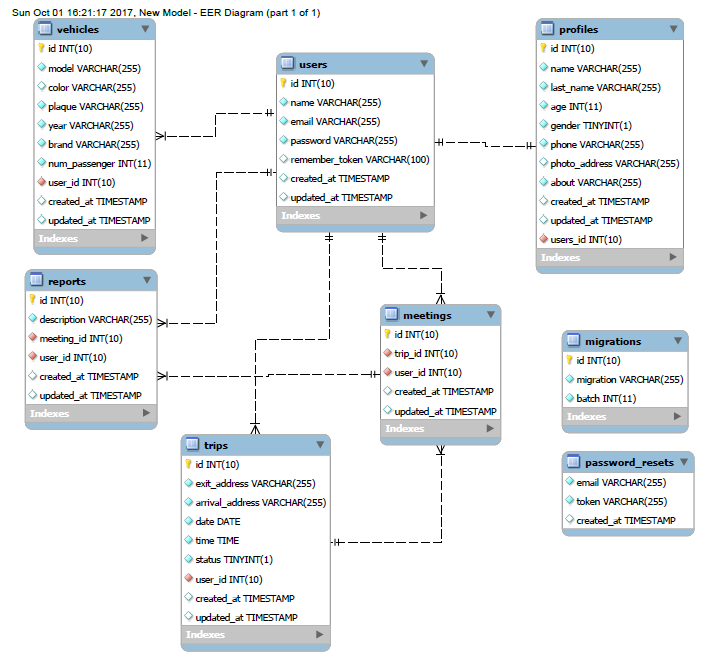
# DIAGRAMA DE SEQUÊNCIA

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.

# 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 6 – 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 @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;

SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;

SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='TRADITIONAL,ALLOW\_INVALID\_DATES';

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

-- Schema mydb

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

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

-- Schema mydb

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

CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;

USE `mydb` ;

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

-- Table `mydb`.`users`

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

CREATE TABLE IF NOT EXISTS `mydb`.`users` (

`id` INT(10) UNSIGNED NOT NULL,

`name` VARCHAR(255) COLLATE 'utf8mb4\_unicodÿÿÿÿÿÿe\_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' NULL DEFAULT NULL,

`created\_at` TIMESTAMP NULL DEFAULT NULL,

`updated\_at` TIMESTAMP NULL DEFAULT NULL,

PRIMARY KEY (`id`),

INDEX `users\_id\_index` (`id` ASC))

ENGINE = InnoDB;

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

-- Table `mydb`.`trips`

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

CREATE TABLE IF NOT EXISTS `mydb`.`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,

`status` TINYINT(1) NOT NULL,

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

`created\_at` TIMESTAMP NULL DEFAULT NULL,

`updated\_at` TIMESTAMP NULL DEFAULT NULL,

PRIMARY KEY (`id`),

INDEX `trips\_user\_id\_foreign` (`user\_id` ASC),

CONSTRAINT `trips\_user\_id\_foreign`

FOREIGN KEY (`user\_id`)

REFERENCES `mydb`.`users` (`id`))

ENGINE = InnoDB;

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

-- Table `mydb`.`meetings`

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

CREATE TABLE IF NOT EXISTS `mydb`.`meetings` (

`id` INT(10) UNSIGNED 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,

PRIMARY KEY (`id`),

INDEX `meetings\_trip\_id\_foreign` (`trip\_id` ASC),

INDEX `meetings\_user\_id\_foreign` (`user\_id` ASC),

CONSTRAINT `meetings\_trip\_id\_foreign`

FOREIGN KEY (`trip\_id`)

REFERENCES `mydb`.`trips` (`id`),

CONSTRAINT `meetings\_user\_id\_foreign`

FOREIGN KEY (`user\_id`)

REFERENCES `mydb`.`users` (`id`))

ENGINE = InnoDB;

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

-- Table `mydb`.`migrations`

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

CREATE TABLE IF NOT EXISTS `mydb`.`migrations` (

`id` INT(10) UNSIGNED NOT NULL,

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

`batch` INT(11) NOT NULL,

PRIMARY KEY (`id`))

ENGINE = InnoDB;

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

-- Table `mydb`.`password\_resets`

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

CREATE TABLE IF NOT EXISTS `mydb`.`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;

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

-- Table `mydb`.`profiles`

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

CREATE TABLE IF NOT EXISTS `mydb`.`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' NULL DEFAULT NULL,

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

`created\_at` TIMESTAMP NULL DEFAULT NULL,

`updated\_at` TIMESTAMP NULL DEFAULT NULL,

`users\_id` INT(10) UNSIGNED NOT NULL,

PRIMARY KEY (`id`),

INDEX `fk\_profiles\_users1\_idx` (`users\_id` ASC),

CONSTRAINT `fk\_profiles\_users1`

FOREIGN KEY (`users\_id`)

REFERENCES `mydb`.`users` (`id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

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

-- Table `mydb`.`reports`

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

CREATE TABLE IF NOT EXISTS `mydb`.`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,

PRIMARY KEY (`id`),

INDEX `reports\_meeting\_id\_foreign` (`meeting\_id` ASC),

INDEX `reports\_user\_id\_foreign` (`user\_id` ASC),

CONSTRAINT `reports\_meeting\_id\_foreign`

FOREIGN KEY (`meeting\_id`)

REFERENCES `mydb`.`meetings` (`id`),

CONSTRAINT `reports\_user\_id\_foreign`

FOREIGN KEY (`user\_id`)

REFERENCES `mydb`.`users` (`id`))

ENGINE = InnoDB;

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

-- Table `mydb`.`vehicles`

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

CREATE TABLE IF NOT EXISTS `mydb`.`vehicles` (

`id` INT(10) UNSIGNED NOT NULL,

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

`color` VARCHAR(255) COLLATE 'utf8mb4\_unicode\_ci' NULL 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,

PRIMARY KEY (`id`),

INDEX `vehicles\_user\_id\_foreign` (`user\_id` ASC),

CONSTRAINT `vehicles\_user\_id\_foreign`

FOREIGN KEY (`user\_id`)

REFERENCES `mydb`.`users` (`id`))

ENGINE = InnoDB;

SET SQL\_MODE=@OLD\_SQL\_MODE;

SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;

SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;--

ALTER TABLE `migrations`

MODIFY `id` int(10) UNSIGNED NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=9;

--

-- AUTO\_INCREMENT for table `phones`

--

ALTER TABLE `phones`

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

--

-- AUTO\_INCREMENT for table `profiles`

--

ALTER TABLE `profiles`

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

--

-- 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=4;

--

-- Constraints for dumped tables

--

--

-- Constraints for table `phones`

--

ALTER TABLE `phones`

ADD CONSTRAINT `phones\_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 `trips`

--

ALTER TABLE `trips`

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

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

# 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 anexo 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 agendamento, 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. Informar o usuário e a senha; |
| **Critérios de Êxito** | Verificar e-mail e senha se corresponde e logar no sistema, caso esteja errado pedir para tentar de novo. |
| **Resultados** | Corretos, tanto para senhas correspondentes e não correspondentes. |

|  |  |
| --- | --- |
| **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** |  |

|  |  |
| --- | --- |
| **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 o numero correspondente da nota para a pessoa; 4. Clicar no botão ... |
| **Critérios de Êxito** | Verificar se foi cadastrada com sucesso a nota da avaliação. |
| **Resultados** |  |

|  |  |
| --- | --- |
| **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** |  |

|  |  |
| --- | --- |
| **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** |  |

|  |  |
| --- | --- |
| **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** |  |

|  |  |
| --- | --- |
| **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** |  |

|  |  |
| --- | --- |
| **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** |  |

|  |  |
| --- | --- |
| **Caso Nº** | CT09 – Visualizar histórico de viagens realizadas |
| **Objetivo do Teste** | Verificar se o usuário consegue visualizar as viagens que foi 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** |  |

|  |  |
| --- | --- |
| **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... |
| **Resultados** |  |

**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 de novembro com pessoas que não fizeram parte do desenvolvimento do aplicativo e que possuem pouca familiaridade com aplicativos. Todos os critérios foram aprovados.

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)