UNIVERSIDADE ESTADUAL DE MONTES CLAROS

Centro de Ciências Exatas e Tecnológicas

Curso de Bacharelado em Sistemas de Informação

Felipe Pereira Quadros Maia

DESENVOLVIMENTO DE SISTEMA WEB PARA PLANTÃO DE VENDAS DE EMPRESAS DE LOTEAMENTO IMOBILIÁRIO

Montes Claros – MG

Outubro de 2016

**Felipe Pereira Quadros Maia**

**DESENVOLVIMENTO DE SISTEMA WEB PARA PLANTÃO DE VENDAS DE EMPRESAS DE LOTEAMENTO IMOBILIÁRIO**

**Monografia apresentada ao Curso de Bacharelado em Sistemas de Informação, da Universidade Estadual de Montes Claros como exigência para obtenção do grau de Bacharel em Sistemas de Informação.**

Orientador: **Prof.ª Sônia Beatriz de Oliveira e Silva Maia, MESTRE.**

**Montes Claros – MG**

**Outubro de 2016**

**Felipe Pereira Quadros Maia**

**DESENVOLVIMENTO DE SISTEMA WEB PARA PLANTÃO DE VENDAS DE EMPRESAS DE LOTEAMENTO IMOBILIÁRIO**

**Monografia apresentada ao Curso de Bacharelado em Sistemas de Informação, da Universidade Estadual de Montes Claros como exigência para obtenção do grau de Bacharel em Sistemas de Informação.**

**Montes Claros, xx de Outubro de 2016.**

Orientador: **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Prof.ª Sônia Beatriz de Oliveira e Silva Maia, MESTRE**

**Universidade Estadual de Montes Claros**

Membros:

**Montes Claros – MG**

**Outubro de 2016**

**AGRADECIMENTOS**

Gostaria de agradecer primeiramente a minha família. Sem meu pai José de Quadro Maia e minha mãe Naldi Pereira Quadros, nada disso seria possível. Gostaria de agradecer a todos os meus amigos que estiveram comigo na jornada da faculdade, em especial aos que fizeram parte do Centro Acadêmico e da Empresa Júnior INFOBITS, a qual me deu o arcabouço necessário para entrar no mercado de trabalho e iniciar a construção de inúmeros sonhos. À minha namorada Ana Luisa Araújo pelo apoio e compreensão. À minha professora e orientadora Sônia Beatriz de Oliveira e Silva Maia pela **enorme** paciência e dedicação durante toda a condução deste trabalho. Enfim, a todos os meus **professores**, colegas, familiares e amigos que de alguma maneira contribuíram para este processo.

***Dedico esta monografia à minha família consanguínea e não consanguínea, que sempre esteve presente em todos os momentos importantes da minha vida.***

**RESUMO**

O presente trabalho aborda o desenvolvimento de um sistema *web* para gestão de um plantão de vendas para empresa de loteamento. O sistema desenvolvido permitirá a gestão dos *status* dos lotes vendidos e disponíveis bem como visualização dos preços de cada lote, além de proporcionar transparência de valores para os corretores e funcionários, bem como o acesso *on-line* para o sócio proprietário. A metodologia de desenvolvimento é realizada de acordo com a metodologia ágil FDD (*Feature Driven Delopment*), utilizando a ferramenta Astah Community para construção dos diagramas UML, Laravel 5.1 para programação *back-end* e html,css e js para *front-end*, e a partir dessas tecnologias foi criada a solução para o problema.

**Palavras chave**: Loteamento, Plantão de Vendas, Modelagem de Sistemas, UML, FDD.

**ABSTRACT**

**LISTA DE ABREVIATURAS E SIGLAS**

|  |  |
| --- | --- |
| **UML**  **ERD** | ***Unified Modeling Language***  ***Entity Relationship Diagram*** |
| **DFD**  **CRUD**  **HTML** | ***Data Flow Diagram***  ***Create, Read, Update, Delete***  ***HyperText Markup Language*** |
| **SGBD**  **FDD** | ***Sistema de Gerenciamento de Banco de Dados***  ***Feature Driven Development*** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**SUMÁRIO**

[1 INTRODUÇÃO 12](#_Toc464313479)

[2 FUNDAMENTAÇÃO TEÓRICA 14](#_Toc464313480)

[2.1 Sistemas de informação 14](#_Toc464313481)

[2.1.1 Sistemas de Informações Gerenciais 15](#_Toc464313482)

[2.2 Engenharia de software 15](#_Toc464313483)

[*2.3* *Metodologia Ágil* 16](#_Toc464313484)

[2.3.1 Feature Drive Development (FDD) 16](#_Toc464313485)

[2.4 Usabilidade na web 17](#_Toc464313486)

[2.5 Linguagem de modelagem unificada (UML) 18](#_Toc464313487)

[2.5.1 Diagrama de Caso de Uso 18](#_Toc464313488)

[2.5.2 Diagrama de Sequência 19](#_Toc464313489)

[2.5.3 Diagrama de Classes 19](#_Toc464313490)

[2.6 Ferramentas Case 19](#_Toc464313491)

[2.6.1 Astah Community 20](#_Toc464313492)

[2.7 HTML 20](#_Toc464313493)

[2.8 CSS 21](#_Toc464313494)

[2.9 PHP 21](#_Toc464313495)

[2.10 Banco de Dados – MySQL 22](#_Toc464313496)

[2.11 Javascript 23](#_Toc464313497)

[2.12 Jquery 23](#_Toc464313498)

[2.13 Laravel 5.1 24](#_Toc464313499)

[3 TRABALHO DESENVOLVIDO 25](#_Toc464313500)

[3.1 Escopo do produto 26](#_Toc464313501)

[3.1.1 Nome do produto e seus principais componentes 27](#_Toc464313502)

[3.2 Concepção e Planejamento 28](#_Toc464313503)

[3.2.1 Modelo Abrangente 28](#_Toc464313504)

[3.2.2 Construir a Lista de Funcionalidades 30](#_Toc464313505)

[3.2.2.1 Lista de Áreas de Negócio 31](#_Toc464313506)

[3.2.2.2 Lista de Áreas de Negócio 31](#_Toc464313507)

[3.2.3 Planejar por Funcionalidade 32](#_Toc464313508)

[3.3 Construção 34](#_Toc464313509)

[3.3.1 Detalhar Por Funcionalidade 34](#_Toc464313510)

[3.3.1.1 Cadastro de Usuário Administrador 34](#_Toc464313511)

[3.3.1.2 Login no Sistema 35](#_Toc464313512)

[3.3.1.3 Cadastro de Empreendimento 37](#_Toc464313513)

[3.3.1.4 Cadastro de Quadra 38](#_Toc464313514)

[3.3.1.5 Cadastro de Lote 40](#_Toc464313515)

[3.3.1.6 Cadastro de Corretor 42](#_Toc464313516)

[3.3.1.7 Selecionar corretor que efetuou a venda 44](#_Toc464313517)

[3.3.1.8 Alterar valor do lote 46](#_Toc464313518)

[3.3.1.9 Alterar porcentagem do corretor 48](#_Toc464313519)

[3.3.1.10 Alterar status do lote 50](#_Toc464313520)

[3.3.1.11 Relatório de Vendas 52](#_Toc464313521)

[3.3.1.12 Relatório de Vendas por Corretor 54](#_Toc464313522)

[3.3.1.13 Relatório de Lotes Vendidos e Disponíveis 56](#_Toc464313523)

[3.4 Construir por Funcionalidade 57](#_Toc464313524)

[3.4.1 Interface de Login 59](#_Toc464313525)

[3.4.2 Interface de Cadastro de Cliente 60](#_Toc464313526)

[3.4.3 Interface de Funcionários 61](#_Toc464313527)

[3.4.4 Interface de Salvar Funcionário 63](#_Toc464313528)

[3.4.5 Interface de Corretores 64](#_Toc464313529)

[3.4.6 Interface de Empreendimentos 65](#_Toc464313530)

[3.4.7 Quadras/lotes 66](#_Toc464313531)

[3.4.8 Vendas 68](#_Toc464313532)

[3.4.9 Usuários e sistemas externos 69](#_Toc464313533)

[4 RESULTADOS OBTIDOS 70](#_Toc464313534)

[5 CONCLUSÃO 72](#_Toc464313535)

[REFERÊNCIAS 73](#_Toc464313536)

# 1 INTRODUÇÃO

A empresa GrandMinas atua no mercado imobiliário desde 1997 e conta com vasta experiência em criar e vender loteamentos em São Paulo e Minas Gerais e é composta por profissionais das mais diversas áreas, desde corretores, engenheiros, administradores, *marketing*, entre outros. Os plantões de vendas dos loteamentos da GrandMinas são gerenciados de forma **arcaica** utilizando papel e caneta sendo um gerenciamento confuso, suscetível a erros e com pouca transparência para os envolvidos no processo de venda no plantão de vendas.

O processo de venda no local do plantão de venda deve ser transparente para os corretores, de fácil utilização para os funcionários e com relatórios claros para o sócio proprietário da GrandMinas. Pensando nisso, este trabalho teve como principal objetivo suprir a necessidade de gerenciamento do processo de venda no momento do plantão de vendas dos loteamentos da empresa GrandMinas, através de um sistema *web* que centraliza, organiza, gera relatórios e deixa transparente os valores dos lotes, o *status* daquele lote e a transparência da porcentagem correta para cada corretor.

Trata-se de um sistema *web* com boa usabilidade e projetado de forma que permita que os usuários do sistema da GrandMinas possam visualizar de forma clara valores e porcentagens da venda de cada lote de modo a agilizar e gerenciar o processo de venda no momento de um plantão de vendas.

Como o sistema está hospedado em servidores *web* os dados gerados podem ser acessados em qualquer parte do mundo, facilitando assim o trabalho do sócio proprietário que em poucas ocasiões se encontra em todos os plantões de vendas.

O tema deste trabalho é o desenvolvimento de um sistema *web* para plantão de vendas de empresas de loteamento imobiliário. Como problema, tem-se a falta de transparência da porcentagem da venda do corretor e o *status* atualizado de um lote se está vendido ou disponível, e o questionamento: a implementação de um sistema *web* para gerenciamento de plantões de venda de loteamentos imobiliários facilitará o gerenciamento e a transparência de um plantão de vendas?

O objetivo geral deste trabalho é o desenvolvimento de sistema *web* para gerenciamento de plantão de vendas de loteamento imobiliário. Através do sistema é possível cadastrar os lotes e os respectivos valores, alterar o *status* da venda entre disponível, vendido e reservado, e calcular o valor da porcentagem do lote vendido por um corretor.

Seus objetivos específicos são desenvolver modelos de módulos que permitam:

* Inclusão, exclusão e edição de empreendimentos;
* Inclusão, exclusão e edição de funcionários;
* Inclusão, exclusão e edição de corretores;
* Inclusão, exclusão e edição de vendas;
* Inclusão, exclusão e edição de quadras;
* Inclusão, exclusão e edição de lotes e seu valor;
* Disponibilização de relatório de venda por corretor;
* Disponibilização de relatório de entradas financeiras de todos os lotes
* Disponibilização de relatório de lotes disponibilizados/vendidos.
* Auditoria

Este trabalho foi dividido em 6 capítulos, promovendo uma melhor organização de suas etapas. O capítulo 2 trata da fundamentação teórica do projeto, que aborda conceitos relativos ao desenvolvimento do trabalho. O capítulo 3, por sua vez, aborda questões sobre o trabalho desenvolvido e sobre os métodos utilizados para o desenvolvimento do sistema. Da mesma forma, o capítulo 4 traz os resultados obtidos com o projeto. No capítulo 5 são apresentadas às conclusões, seguidas das referências na elaboração desse presente trabalho.

# 2 FUNDAMENTAÇÃO TEÓRICA

A seguir estão descritos os principais conceitos relacionados ao desenvolvimento do projeto.

## Sistemas de informação

Conforme Laudon e Laudon (2007, p. 9, grifo do autor), “um **sistema de informação** pode ser definido tecnicamente como um conjunto de componentes inter-relacionados que coletam (ou recuperam), processam, armazenam e distribuem informações destinadas a apoiar a tomada de decisões, a coordenação e o controle de uma organização”.

De acordo com Rezende (2005), os Sistemas de Informação podem ser classificados, levando em consideração o suporte à decisão em:

* Sistemas de Informação Operacionais: Auxiliam a tomada de decisão do setor técnico empresarial, através do processamento de informações e operações que fazem parte da rotina da organização;
* Sistemas de Informações Gerenciais: Permite a tomada de decisão da equipe gerente da organização, através do processamento de informações e dados agrupados das operações;
* Sistemas de Informações Estratégicas: Transforma as informações operacionais em informações estratégicas. Com a utilização desse tipo de sistema, é levado em consideração o meio ambiente interno e externo à organização.

Ainda segundo Laudon e Laudon (2007) apontam que a compreensão de um sistema de informação passa por algumas dimensões mais amplas, como, a capacidade de prover soluções para desafios em um ambiente empresarial.

Com o intuito de ajudar as empresas na otimização dos seus processos e na tomada de decisões será utilizado de um Sistema de Informação Gerencial no presente trabalho.

### Sistemas de Informações Gerenciais

Segundo Laudon e Laudon (2009, p. XIV), “existe um mundo de negócios completamente novo utilizando novas tecnologias para gerenciar e organizar. O que faz da área de sistemas de informações gerencias a mais excitante de estudo nas escolas de administração é a mudança continua em tecnologia, gestão e processos de negócios“.

Rezende (2005) diz que os Sistemas Gerencias de Informação são aplicados para que haja o processamento de grupos de dados das operações com o intuito de sintetizá-los para aplicação no auxílio em tomadas de decisão.

Sendo assim, serão adotadas práticas de engenharia de *software* para a construção coerente de um Sistema de Informação Gerencial.

## Engenharia de software

Segundo Pressman (2011, p. 39), “Engenharia de Software é o estabelecimento e o emprego de sólidos princípios de engenharia de modo a obter software de maneira econômia, que seja confiável e funcione de forma eficiente em máquinas reais”.

De acordo com Rezende (2005), a engenharia de *software* está em paralelo com os sistemas de informação de maneira a auxiliar as organizações na tomada de decisões sob o foco de seu negócio empresarial ou atividade pública.

Para Lobo (2008), temos por função definir métodos sistemáticos para a criação dos sistemas com a finalidade de otimizar e amadurecer técnicas e ferramentas para todo o ambiente de desenvolvimento.

Ainda segundo Lobo (2008, p. 8), “Engenharia de Software busca definir atividades e técnicas a serem utilizadas na criação do software, oferecendo modelos, padrões, arquiteturas, métodos e processos que serão bem-vindos ao ambiente de desenvolvimento ”.

Segundo Pressman e Maxim (2016), os engenheiros de *software* devem saber gerenciar mudanças, comunicar-se com os envolvidos e usar ferramentas adequadas nas situações apropriadas com a finalidade intrínseca de buscar a qualidade do *software*.

## *Metodologia Ágil*

Segundo Mansur (2007, p. 167), “Os requisitos de projetos mudam em função da velocidade de mudança nas regras do negócio e do tempo de entrega do software. Em geral, a agilidade das mudanças das regras do negócio impacta o tempo de entrega do software através de mudanças nos requisitos”.

Ainda para Mansur (2007) o ambiente de negócios é muito dinâmico e o desenvolvimento de *software* é muito longo se comparado ao de negócios, sendo assim, as metodologias ágeis são uma resposta que possibilita a redução do tempo na entrega de *softwares*.

Para Lobo (2008, p. 43) as principais características da metodologia ágil são:

* **Rápida e objetiva:** Não é burocrática. Está com prioridade nas funcionalidades do *software*.
* **Leve:** Utiliza regras e métodos diretos, não burocráticos.
* **Iterações curtas:** Utiliza iterações de curto de período de tempo (de uma a quatro semanas).
* **Diminui riscos do projeto:** Possui iterações curtas e documentação leve, permitindo assim qualquer mudança necessária para atender às necessidades do cliente.
* **Implementa versões do *software*:** Define e controla uma versão do *software* para cada iteração do projeto.
* **Visa prioridades:** Enfatiza as prioridades do projeto. O projeto será desenvolvido com foco nas prioridades.
* **Documentação leve e eficiente:** Utiliza protótipos, modelos e diagramas para representar o *software*, através de uma ótica direta e ágil.

Para desenvolvimento desse trabalho de conclusão de curso será utilizar-se-á a metodologia ágil denominada *Feature Driven Development* (FDD) que será descrita a seguir.

## Feature Drive Development (FDD)

Segundo Massari (2016, p. 257), “o Feature-Driven Development ou desenvolvimento orientado a funcionalidades é uma metodologia ágil que consiste em:

* desenhar um protótipo do produto;
* montar uma lista de funcionalidades desse produto;
* planejar por funcionalidades;
* desenvolver por funcionalidade;
* entregar por funcionalidade.”

Wazlawick (2013), afirma que o FDD é um método que visa focar no uso de orientação de objetos e tem duas grandes fases definidas onde constam a **Concepção e o planejamento** e **Construção**. Na fase de Concepção e planejamento consiste em planejar, discutir e pensar um pouco antes de começar a construir, aconselha-se em torno de duas semanas. Já a fase de Construção é onde tem-se o desenvolvimento iterativo do produto com vários ciclos desenvolvendo funcionalidades onde em que ciclo chega de uma a duas semanas.

Será utilizado nesse projeto o FDD para guiar o desenvolvimento do *software* e para guiar a modelagem utilizaremos UML.

## Usabilidade na web

Nielsen e Loranger (2007) dizem que os usuários estão menos tolerantes em relação à complexidade dos *sites* e páginas da *Web*. A usabilidade nunca esteve em um patamar tão importante, sendo que a falha em um projeto em decorrência a ela significa negócios perdidos. Ainda de acordo com os autores, a usabilidade funciona porque revela como o mundo funciona.

De acordo com Andrade (2007), a usabilidade permite que o homem use os objetos técnicos envolvidos de maneira transparente e automática, refletindo não apenas como uma extensão do seu corpo, mas também como uma quebra de horizontes, uma desconexão momentânea da realidade externa e uma extensão do seu mundo físico.

Nielsen e Loranger (2007, p. xvi) também definem a usabilidade como:

Um atributo de qualidade relacionado à facilidade do uso de algo. Mais especificamente, refere-se à rapidez com que os usuários podem aprender a usar alguma coisa, a eficiência deles a usá-la o quanto lembram daquilo, seu grau de propensão a erros e o quanto gostam de utilizá-la. Se as pessoas não puderem ou não utilizarem um recurso, ele pode muito bem não existir.

## Linguagem de modelagem unificada (UML)

De acordo com Melo (2010), a UML é uma forma gráfica que será utilizada de forma fácil e expressiva. A facilidade de entendimento de uma imagem é direta e simples, enquanto a uma descrição pode ser extensa. A UML é amplamente adotada devido a sua simplicidade de entendimento do desenvolvedor e do usuário.

Ainda segundo MELO (2010, p. 35), “UML (Unified Modeling Language – Linguagem de Modelagem Unificada) é uma linguagem para especificação, visualização, construção e documentação de artefatos de sistemas de software”.

Segundo BOOCH (2005, p. 5), “A modelagem é uma parte central de todas as atividades que levam à implantação de um bom software. Construímos modelos para comunicar a estrutura e o comportamento desejados do sistema.”

No projeto será listado os Diagramas de Classes, Diagrama de Uso e Diagrama de Sequência.

### Diagrama de Caso de Uso

Rezende (2005) afirma que a UML é a linguagem utilizada para a construção de um diagrama de caso de uso. O Diagrama de Caso de Uso é utilizado para descrever um Modelo de Caso de Uso, seus atores, relacionamentos, generalizações, associações e dependências entre elementos.

Rezende (2005, p. 205) lista as seguintes definições:

O ator é alguém ou alguma coisa que interage com o sistema, envia ou recebe mensagem, troca informações.[...]

A associação entre casos de uso pode ser: inclusão, generalização e extensão. A inclusão ocorre quando há uma parte do comportamento que é semelhante em mais de um caso de uso.

A generalização é quando tem um caso de uso semelhante ao outro e a extensão é semelhante a generalização mas contém mais regras, podendo acrescentar comportamentos adicionais.

A interação entre casos de uso pode ser: comunicação, extensão e uso. A comunicação é a conexão de caminhos. A extensão mostra comportamentos de exceção do caso de uso, e o Uso é quando o caso de uso tem comportamento comum que pode ser modelado e usado por outros.

### Diagrama de Sequência

De acordo com Rezende (2005), o diagrama de sequência é um diagrama de interação para demonstrar troca de mensagens. O diagrama demonstra a interação entre objetos de acordo com uma determinada sequência de mensagens, isso implica na demonstração de troca de mensagens entre objetos. É composto por um eixo horizontal com a sequência de objetos e outro vertical com o momento da mensagem.

### Diagrama de Classes

Segundo MANINO (2014, p. 157), “Os diagramas de classe contêm classes (grupos de objetos), associações (relacionamentos binários) entre classes e características dos objetos (atributos e operações)”.

SAMPAIO (2007, p. 25) lista os tipos de relacionamento entre classes, que são:

**1) Dependência:** Quando a classe “A” depende da classe “B”, no sentido em que alterações em “B” provocarão alterações em “A”.

**2) Associação:** Quando dois objetos estão associados entre si, no sentido de propriedade. Um objeto de classe “A” está associado a um objeto de classe “B” porque possui uma variável do tipo “B”.

**3) Generalização:** Também conhecido como “Gen-spec”, determina um relacionamento de herança entre classes.

**4) Realização:** Quando a classe “A” implementa a interface “I”.

## 2.6 Ferramentas Case

Segundo Meireles(2004), são inúmeras as ferramentas que auxiliam a automatização de processos para analistas e programadores em várias fases e atividades.

Ainda segundo MEIRELES (2004, p. 74), “As ferramentas CASE (Computer Aided Software Engineering – Engenharia de Software auxiliada por computador) permitem desenvolver as atividades de análise e projeto com auxílio computadorizado, com a vantagem de criar um ambiente integrado de informações.”

### 2.6.1 Astah Community

“ASTAH é um editor UML leve, integrado com ERD, DFD, CRUD e Mind Mapping que são recursos para desenvolvedores de software.” (VISION, 2009, tradução nossa)

O Astah community compreende todos os diagramas da UML 2.1, que são:

A) Diagrama de Classe

B) Diagrama de Caso de Uso

C) Diagrama de estado de máquina

D) Diagrama de Atividade

E) Diagrama de Sequência

F) Diagrama de Comunicação

G) Diagrama de Componentes

H) Diagrama de Implantação

I) Diagrama de Estrutura Composta

## 2.7 HTML

Carril (2012, p.3) define que, “HTML é uma linguagem de marcação, utilizada para formatar páginas Web com texto e informações separadamente. Estas páginas são acessíveis em qualquer parte do mundo usando qualquer tipo de browser […]. Estes browsers tem a função de interpretar, formatar e exibir página WEB”.

De forma simplista, Tittle e Noble (2014, p. 10) afirmam que “Se você navegar na web somente por um instante, verá um espaço com informações e conteúdos exibidos de diversas maneiras. […] Independente das informações de uma página web, cada uma delas é criada usando-se HTML. HTML é a argamassa que mantém as páginas unidas na web”.

## 2.8 CSS

De acordo com Jobstraibizer (2009, p. 6), o css é importante para que o sites fiquem mais bonitos, leves, limpos e dinâmicos.

Ainda de acordo com Jobstraibizer (2009, p. 6), “O termo CSS, […] é uma linguagem de estilo (ou de formatação) fantástica para a construção do layout de suas páginas ou sites. Permite que projete páginas […] com uma considerável redução de tempo de trabalho.”

Jobstraibizer (2009, p. 6), lista os inúmeros benefícios no uso de CSS para formatar conteúdo de páginas:

1) Total controle do *layout* de vários documentos a partir de uma única folha de estilos;

2) Maior precisão no controle do *layout* como um todo (elementos de texto, imagens, visual da página etc.);

3) Emprego de técnicas de desenvolvimento variadas, sofisticadas e avançadas, como, por exemplo, *pop-ups* flutuantes, uso de fontes diferenciadas, etc.

## 2.9 PHP

Segundo Converse e Park (2003, p. 3), “O php é uma linguagem para a criação de scripts para a Web do lado servidor embutidos em HTML, cujo código fonte é aberto, e que é compatível com os mais importantes servidores Web”.

Alguns motivos para utilização do php, de acordo com Converse e Park (2003, p. 3), são que “ele é livre, tem código fonte aberto, é repleto de recursos, serve para diversas plataformas, é estável, rápido, claramente projetado, fácil de aprender e funciona bem com outros produtos”.

## 2.10 Banco de Dados – MySQL

Ramakrishnan e Gehrke (2008, p. 6) afirmam que “Armazenando-se dados em um SGBD em vez de em uma coleção de arquivos do sistema operacional, é possível utilizar os recursos do SGBD para gerenciar dados de uma forma robusta e eficiente.”

Segundo Ramakrishnan e Gehrke (2008, p. 7) as vantagens de um sgbd são:

“**1) Independência de dados:** Os programas aplicativos não devem, idealmente, ser expostos aos detalhes de representação e armazenamento de dados. O SGBD provê uma visão abstrata dos dados que oculta tais detalhes.

**2) Acesso eficiente aos Dados:** Um SGBD utiliza uma variedade de técnicas sofisticadas para armazenar e recuperar dados eficientemente. Este recurso é especialmente importante se os dados são armazenados em dispositivos de armazenamento externos.

**3) Administração de Dados:** Quando diversos usuários compartilham dados, centralizar a administração dos dados pode oferecer melhorias significativas. Profissionais experientes que compreendem a natureza dos dados sendo gerenciados, e como os diferentes grupos de usuários os utilizam, podem ser responsáveis por organizar a representação dos dados para minimizar a redundância e para realizar as sintonizações finas do armazenamento dos dados para garantir uma eficiente recuperação.

**4) Acesso concorrente e recuperação de falhas:** Um SGBD planeja o acesso concorrente aos dados de maneira tal que os usuários podem achar que os dados estão sendo acessados por apenas um único usuário de cada vez. Além disso, o SGBD protege os usuários dos efeitos de falhas no sistema.

**5) Tempo reduzido de Desenvolvimento do Aplicativo:** Obviamente, um SGBD suporte funções importantes que são comuns a várias aplicativos que acessam os dados do SGBD. Isso em conjunto com uma interface de alto nível aos dados, facilita o desenvolvimento rápido de aplicativos.”

Segundo Lobo (2008), o MySQL é um SGBD veloz e multiplataforma que através de sua flexibilidade foi altamente aplicado por todo o mundo. Atualmente é o banco de dados mais popular do mundo e isso se deve a sua facilidade e ao código aberto, podendo alterar de graça.

Muto (2006 , p. 184), nos diz que “uma das grandes vantagens do MySQL é suportar mais de uma dúzia de tipos de dados, suportando ainda funções SQL, através do ANSI SQL. Claro que nem tudo são maravilhas nesse SGBD, faltando principalmente, um eficiente controle de transações”.

## 2.11 Javascript

Prescott (2016) define que javascript, “é a linguagem de script do lado cliente mais amplamente utilizada. Ela é realmente poderosa e dinâmica. […] Linguagens de script do lado do cliente são usadas para tornar seus sites mais interativos, animados e responsivos”.

Ainda segundo Prescott(2016), “você pode usar o javascript para ajudar os usuários a interagirem melhor com páginas da Web. O javascript também pode ser usado para controlar o navegador, comunicar-se de forma assíncrona com o servidor, alterar de forma dinâmica um conteúdo de uma página Web”.

## 2.12 Jquery

Segundo (Documentação oficial, tradução nossa), “jQuery é uma biblioteca Javascript rápida, pequena e rica em recursos. Isso torna as coisas como manipulação de eventos, animação e ajax muito mais simples com uma API fácil de usar que funcionar através de uma variedade de navegadores.”

Silva (2014 ,p. 25) deixa claro que jQuery é “uma maneira simples e fácil de escrever JavaScript colocada ao alcance não só de programadores experientes, mas também de designers e desenvolvedores com pouco conhecimento de programação”.

## 2.13 Laravel 5.1

Segundo Adriel (2015), “Laravel é um framework PHP utilizado para o desenvolvimento web, que utiliza a arquitetura MVC e tem como principal característica ajudar a desenvolver aplicações seguras e **performáticas** de forma rápido, com código limpo e simples”.

Adriel (2015) também diz que, o laravel utiliza o *Blade* excelente *engine* de *template*, para integração com Banco de Dados o Eloquent ORM e versão LTS.

# TRABALHO DESENVOLVIDO

O trabalho foi iniciado com o objetivo de sanar o problema de baixo controle na gestão das vendas dos lotes em um plantão de venda. O primeiro passo foi o levantamento de requisitos que se deu através da observação do funcionamento de um plantão de vendas real da empresa GrandMinas e reuniões com corretores, funcionários e o sócio proprietário da empresa. Ficou definido então que o *software* deverá ter os requisitos funcionais e não funcionais a seguir:

* Deverá ser um sistema *web*;
* Deverá gerenciar mais de um loteamento;
* Apresentar tela de cadastro, listagem e exclusão dos empreendimentos;
* Cadastrar, excluir, listar quadras e lotes relacionadas aos empreendimentos;
* Lotes devem ter Nome e valor;
* Somente administrador pode adicionar funcionários;
* Funcionários podem adicionar corretores;
* Corretores tem acesso somente aos relatórios de lotes vendidos e o financeiro do corretor;
* Terá uma tela de vendas dos lotes. Essa tela mapeará o *status* dos lotes;
* Os *status* dos lotes serão: Disponível, Em negociação, Reservado, Vendido;
* O acesso ao sistema é restrito a administrador, funcionários e corretores;
* O sistema deve calcular a porcentagem correta do valor do lote que irá para o corretor;
* O sistema deve oferecer um relatório financeiro geral com todos os lotes vendidos e o valor total adquirido;
* O sistema deve ranquear o melhor corretor através do total do valor vendido;
* O sistema deve mostrar graficamente a quantidade de lotes/*status*.

Devido a necessidade do cliente em receber as funcionalidades desenvolvidas prontas para uso, definiu-se a metodologia FDD para desenvolvimento do *software*. Após o levantamento de requisitos, foi delineado o modelo abrangente utilizando a ferramenta Astah Community e os diagramas de Caso de Uso, de Classe e de Sequência com o intuito de descrever com mais detalhes e graficamente as funcionalidades e suas possíveis implementações. Ao analisar as prioridades do sistema e os diagramas foi criada uma lista de **funcionalidades** por importância, levando em conta o fator de dependência entre **funcionalidades** e a necessidades das **funcionalidades** pela GrandMinas. A partir desse ponto, foi definido um tempo de entrega das **funcionalidades** de 2 semanas, e deu-se início ao planejamento da **funcionalidade** selecionada para desenvolvimento detalhando-a em tarefas menores para implementação. Nesse ponto se deu a programação da **funcionalidade** até a sua entrega e recomeço desse ciclo de desenvolvimento até o final de toda a lista de **funcionalidades**.

## Escopo do produto

O sistema utilizou de tecnologias novas e padronizadas como o Laravel auxiliando na performance do código e os *frameworks* para *front-end* jQuery e Bootstrap para que o sistema tenha alta usabilidade.

Para gerenciamento do desenvolvimento do sistema foi utilizado a metodologia ágil *Feature Driven Development*. O FDD, é constituído de duas fases e cinco processos listados, em ordem descrita, sendo:

**Fases**:

* Concepção e Planejamento (Prazo do projeto: 1 semana)
* Construção (Iterações de 2 semanas)

**Processos**:

* DMA (Desenvolver Modelo Abrangente)
* CLF (Construir a lista de funcionalidades)
* PPF (Planejar por funcionalidade)
* DPF (Detalhar por funcionalidade)
* CPF (Construir por funcionalidade)

A seguir são apresentadas as características do sistema.

### Nome do produto e seus principais componentes

Nome

<https://pacific-thicket-57678.herokuapp.com/>– Sistema de gestão de plantão de vendas de loteamento.

Principais Componentes

Cadastro

Vendas

Relatórios

#### Missão do produto

Efetuar a gestão do processo de plantão de vendas utilizando um sistema *web* que seja transparente com os valores de porcentagem para os corretores, que organize o *status* das vendas para corretores e funcionários e que seja acessado em qualquer lugar do mundo pelo Sócio Proprietário da GrandMinas.

#### Limites do produto

1. O sistema possuirá um usuário *administrador* cadastrado que terá acesso a todas as funcionalidades do sistema. É através desse usuário que os outros usuários poderão ter sua permissão de acesso ao sistema, alteradas;
2. O usuário corretor terá acesso somente ao relatório de corretor/venda;
3. O sistema não permitirá que funcionários tenham acesso ao relatório geral de vendas do sistema.

#### Benefícios do produto

Os benefícios do produto são apresentados no Quadro 1.

Quadro 1 – BENEFÍCIOS DO PRODUTO

|  |  |  |
| --- | --- | --- |
| Número de ordem | Benefício | Valor para o cliente |
| 1 | Agilidade na alteração de *status* da venda. | Essencial |
|  |  |  |
| 2 | Boa usabilidade. | Desejável |
|  |  |  |
| 3 | *Log* de uso da aplicação. | Desejável |
|  |  |  |
| 4 | Transparência ao demonstrar dados para corretores | Essencial |
|  |  |  |

Fonte: **PRÓPRIA, 2016**

## Concepção e Planejamento

## Modelo Abrangente

Nessa fase do desenvolvimento realiza-se um estudo dirigido sobre o escopo do sistema e seu contexto. Foi escolhido o diagrama de contexto e o diagrama de classes para modelar o sistema. Foram escolhidos esses diagramas devido a clareza de informações para desenvolvimento.

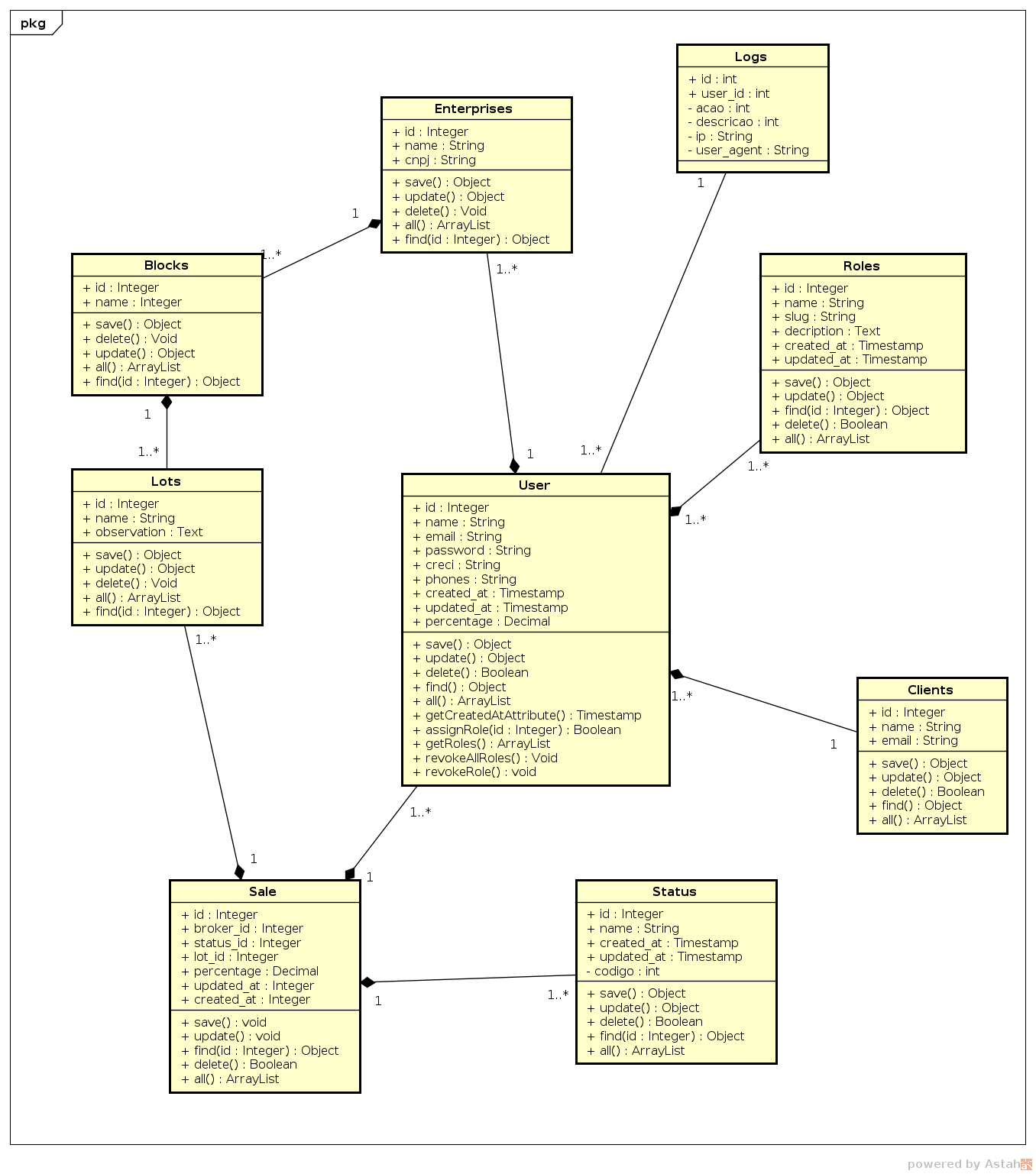
A FIG. 1 mostra o diagrama de contexto do sistema, que foi desenvolvido através do software *Astah Community,* versão 6.8.0.

Figura 1 – Diagrama de contexto



Fonte: **PRÓPRIA, 2016**.

Figura 2 – Diagrama de classes



Fonte: **PRÓPRIA, 2016**.

### Construir a Lista de Funcionalidades

Nessa fase, é criada a lista de funcionalidades para decompor funcionalmente o domínio em áreas de negócio a fim de formar a lista categorizada de funcionalidades. É uma lista mais geral por áreas de negócio do sistema.

### Lista de Áreas de Negócio

Tem-se a seguir a lista das áreas de negócio do sistema. A partir dela surgiram as funcionalidades desenvolvidas.

* Cadastros
* Vendas
* Relatórios

### Lista de Áreas de Negócio

No **Quadro 2** segue a lista de atividades por área de negócio

|  |  |
| --- | --- |
| **Quadro 2 - Atividades por área de negócio** | |
| 1. Cadastros | * 1. Cadastro de Usuário Administrador   2. Cadastro de Corretor   3. Cadastro de Funcionário   4. Cadastro de Empreendimento   5. Cadastro de Quadra   6. Cadastro de Lote   7. Listagem de Corretores   8. Listagem de Funcionários   9. Listagem de Empreendimento   10. Listagem de Quadra   11. Listagem de Lote   12. Exclusão de Corretor   13. Exclusão de Funcionário   14. Exclusão de Empreendimento   15. Exclusão de Quadra   16. Exclusão de Lote |
| 1. Vendas | 2.1) Selecionar Corretor que efetuou a venda  2.2) Alterar Valor do Lote  2.3) Alterar porcentagem do corretor  2.4) Alterar Status do Lote |
| 1. Relatórios | 3.1) Relatório de Vendas  3.2) Relatório de Vendas por Corretor  3.3) Relatório de Lotes Disponíveis/Vendidos |

Fonte: **PRÓPRIA, 2016**

### Planejar por Funcionalidade

Tem-se nessa etapa a ordem de implementação das atividades e o tempo utilizado para as iterações. As atividades são priorizadas de acordo com a dependência entre elas, bem como o prazo do desenvolvimento.

**Prazo de Iterações: 2 semanas**

|  |
| --- |
| **Quadro 3 – Funcionalidades priorizadas e com Prazo** |
| **Início: 28/03/2016 Fim: 08/04/2016** |
| 1. Cadastro de Usuário Administrador 2. *Login* no Sistema 3. Cadastro de Empreendimento 4. Cadastro de Quadra 5. Cadastro de Lote |
| **Início: 09/04/2016 Fim: 22/04/2016** |
| 1. Cadastro de Corretor 2. Selecionar Corretor que efetuou a venda 3. Alterar valor do lote 4. Alterar porcentagem do corretor 5. Alterar *Status* do Lote |
| **Início: 23/04/2016 Fim: 07/05/2016** |
| 1. Relatório de Vendas 2. Relatório de Vendas por Corretor 3. Relatório de Lotes Disponíveis e Vendidos 4. Cadastro de Funcionário 5. Listagem de Funcionário 6. Exclusão de Funcionário |
| **Início: 08/05/2016 Fim: 22/05/2016** |
| 1. Listagem de Empreendimentos 2. Exclusão de Empreendimentos 3. Listagem de Quadras 4. Exclusão de Quadra 5. Listagem de Lotes 6. Exclusão de Lotes |
| **Início: 23/05/2016 Fim: 06/06/2016** |
| 1. Listagem de Corretores 2. Exclusão de Corretores 3. *Logout* do Sistema 4. *Log* de ações da aplicação 5. Recuperar Senha |

Fonte: **PRÓPRIA, 2016**

## Construção

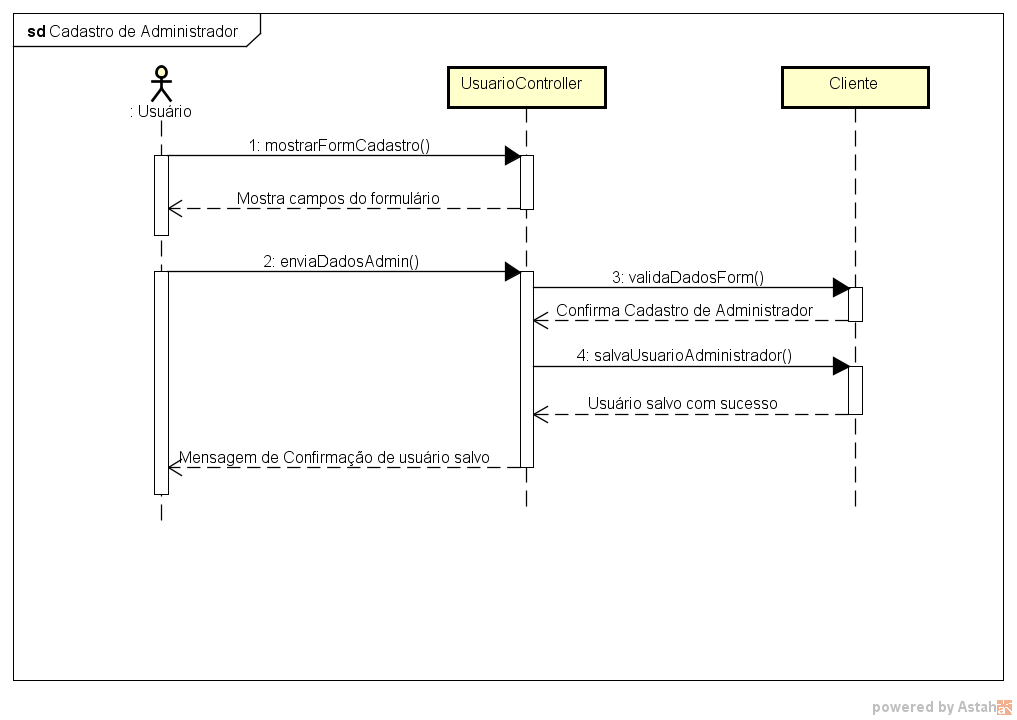
A fase de construção é composta pelas iterações. O primeiro passo de uma iteração é o DFD (Detalhar por Funcionalidade) que consiste em gerar o diagrama de sequência e mostrar o modelo abrangente específico da funcionalidade. O segundo passo é o CPF (Construir por funcionalidade), onde é escrito o código, feito o *build* e gerada a funcionalidade pronta. Esse ciclo é repetido até o fim do desenvolvimento do sistema.

### Detalhar Por Funcionalidade

Nessa fase se detalha a funcionalidade com o diagrama de sequência e, caso necessário, refina-se o modelo abrangente específico a cada funcionalidade através do diagrama de classe.

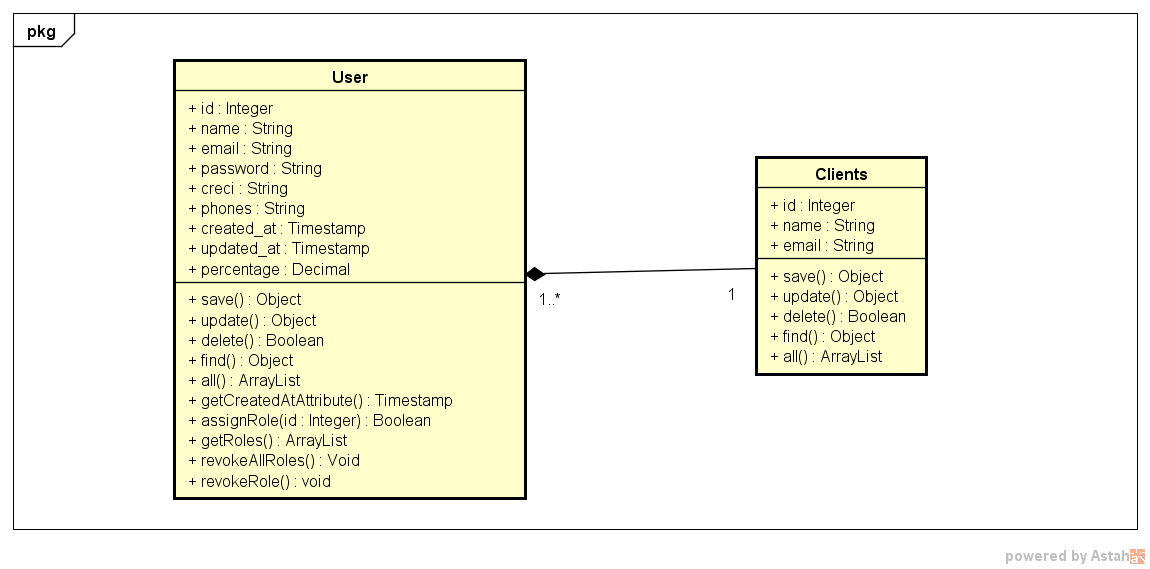
### Cadastro de Usuário Administrador

1. Figura 3 – Diagrama de sequência



Fonte: **PRÓPRIA, 2016**

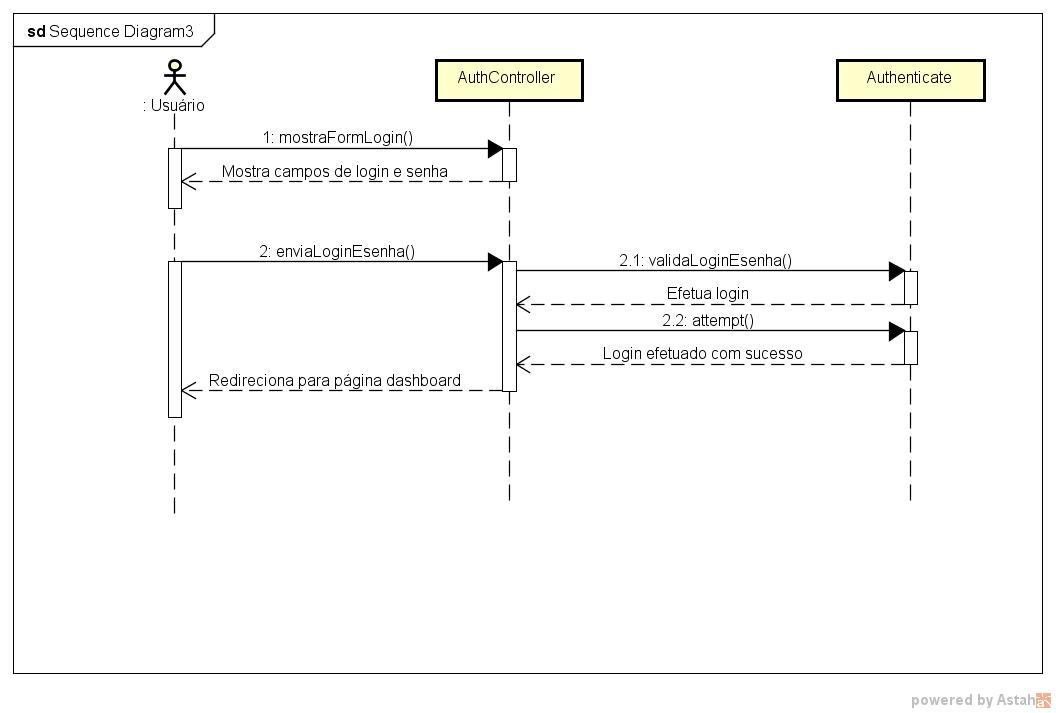
Figura 4 – Diagrama de Classe



Fonte: **PRÓPRIA, 2016**

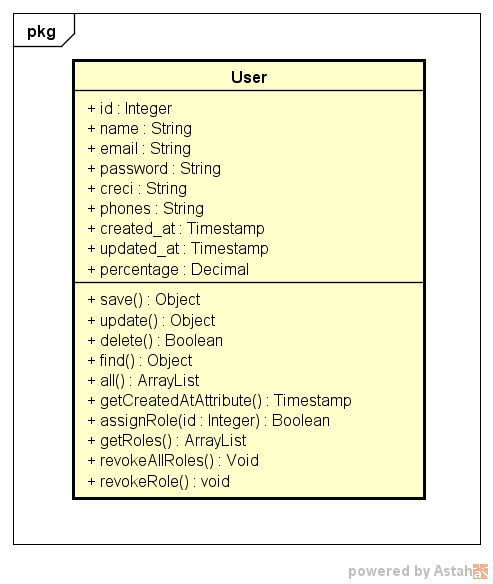
### Login no Sistema

Figura 5 – Diagrama de Sequência



Fonte: **PRÓPRIA, 2016**

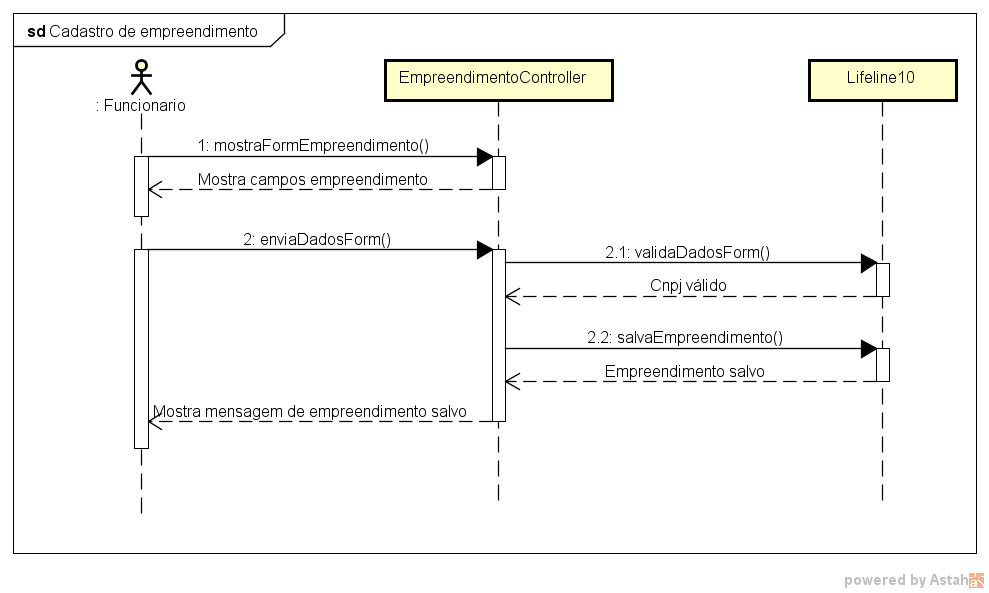
Figura 6 – Diagrama de Classe



Fonte: **PRÓPRIA, 2016**

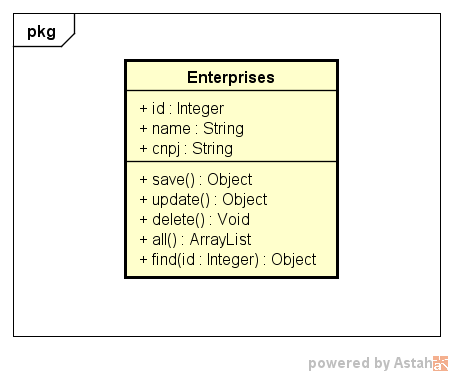
### Cadastro de Empreendimento

Figura 7 – Diagrama de Sequência



Fonte: **PRÓPRIA, 2016**

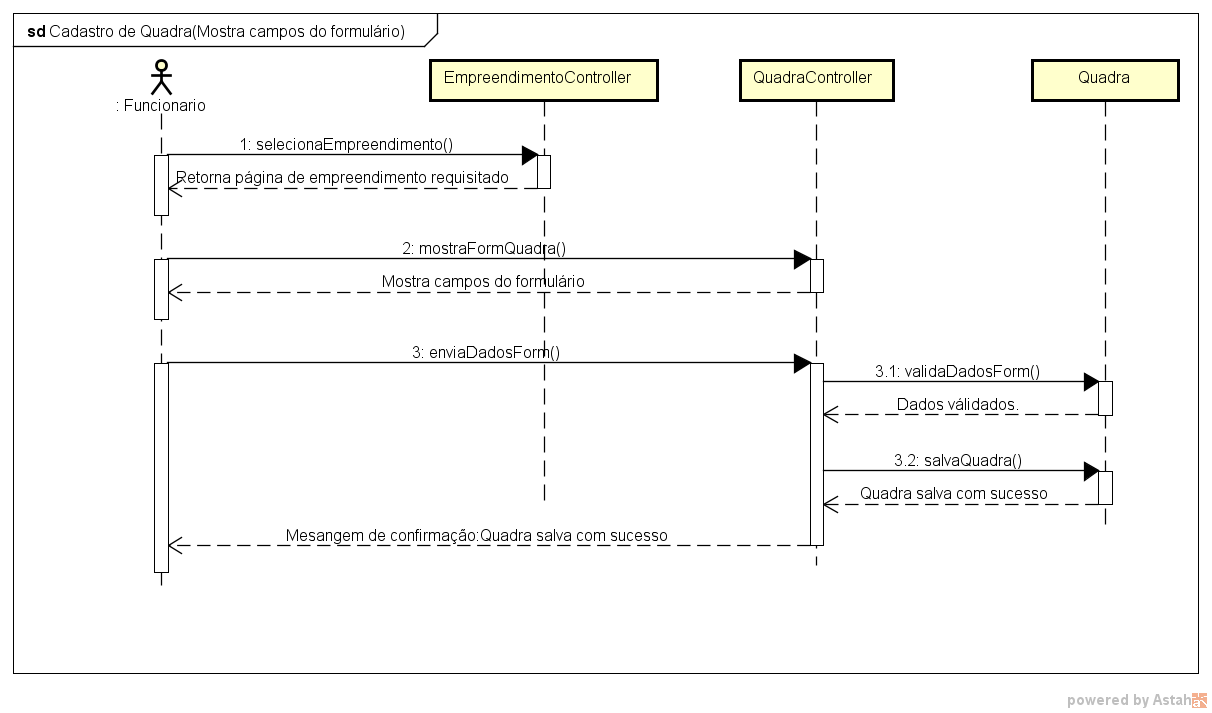
Figura 8 – Diagrama de Classe



Fonte: **PRÓPRIA, 2016**

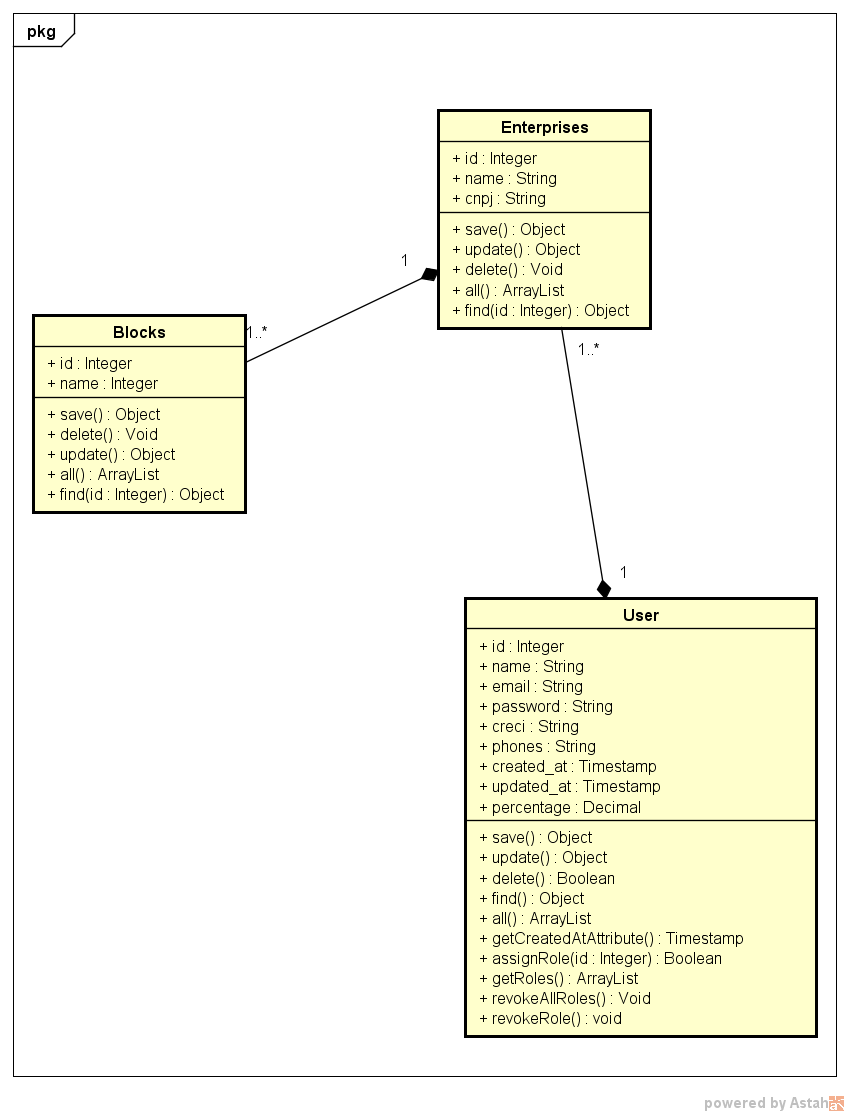
### Cadastro de Quadra

Figura 9 – Diagrama de Sequência



Fonte: **PRÓPRIA, 2016**

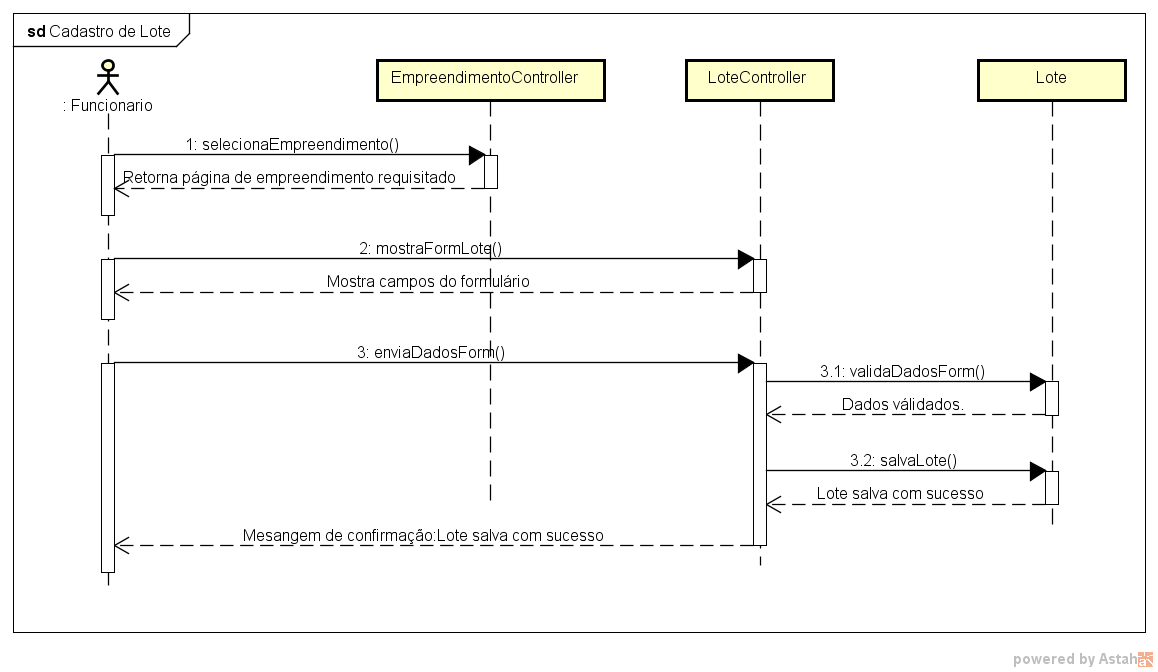
Figura 10 – Diagrama de Classe



Fonte: **PRÓPRIA, 2016**

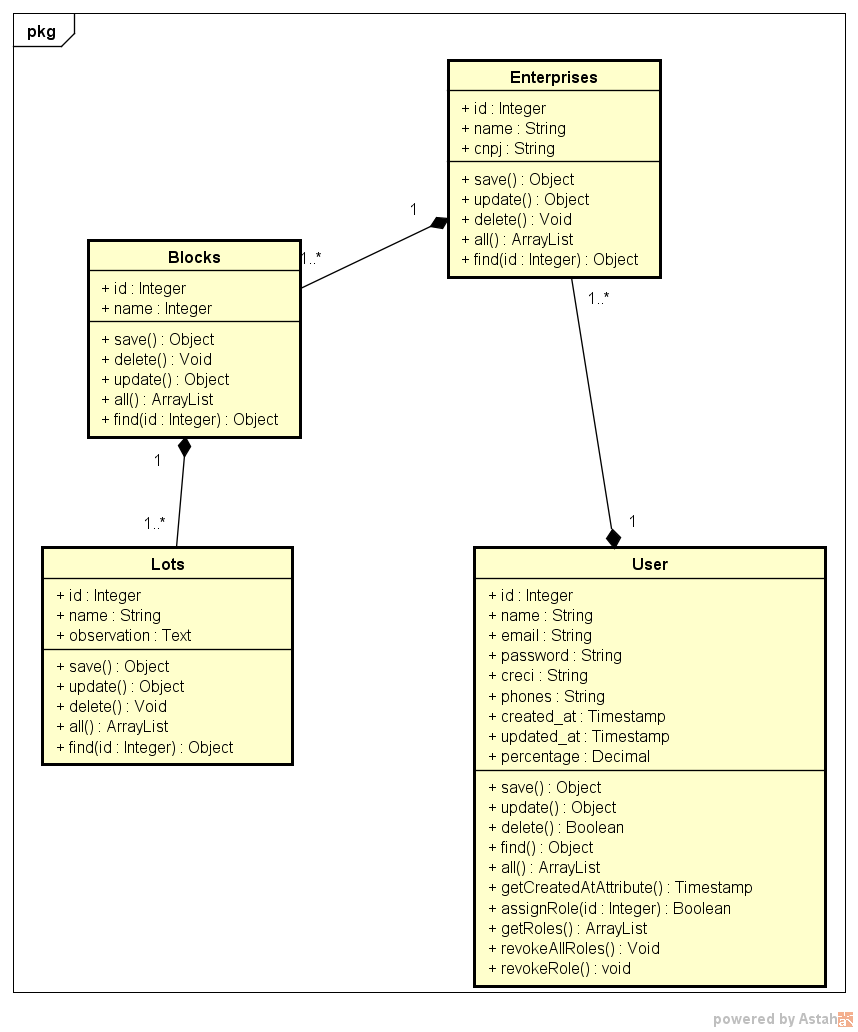
### Cadastro de Lote

Figura 11 – Diagrama de Sequência



Fonte: **PRÓPRIA, 2016**

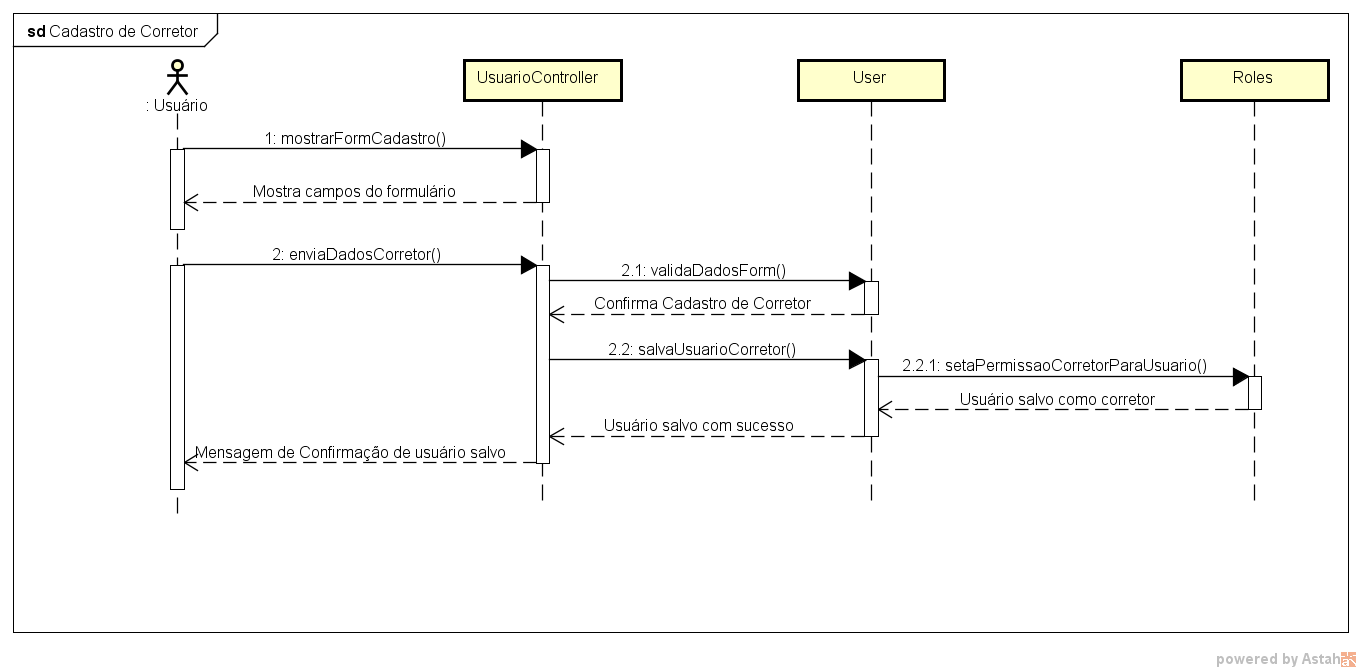
Figura 12 – Diagrama de Classe



Fonte: **PRÓPRIA, 2016**

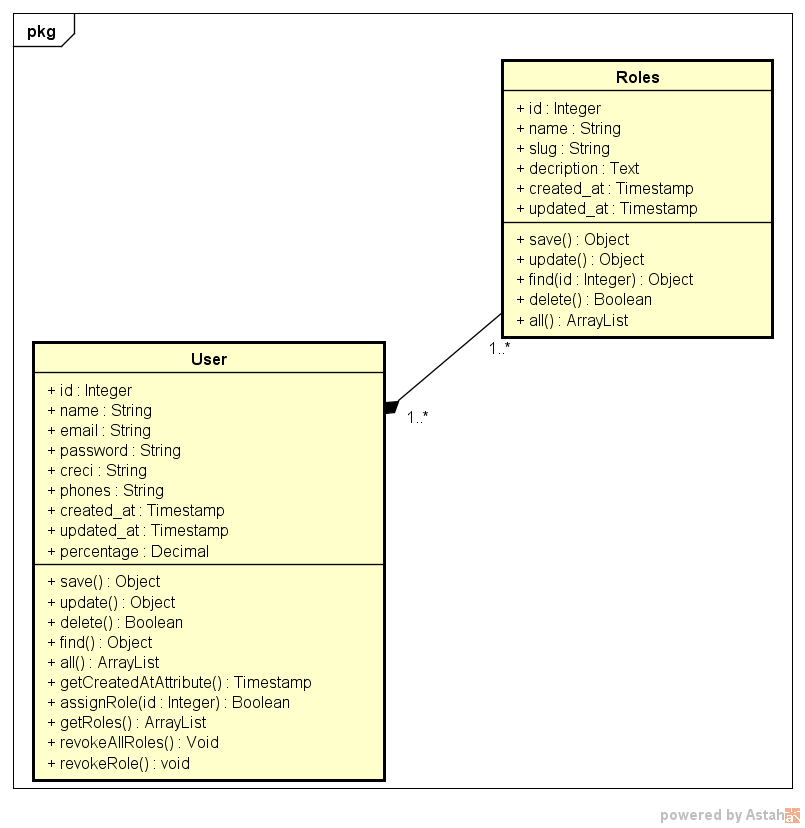
### Cadastro de Corretor

Figura 13 – Diagrama de Sequência



Fonte: **PRÓPRIA, 2016**

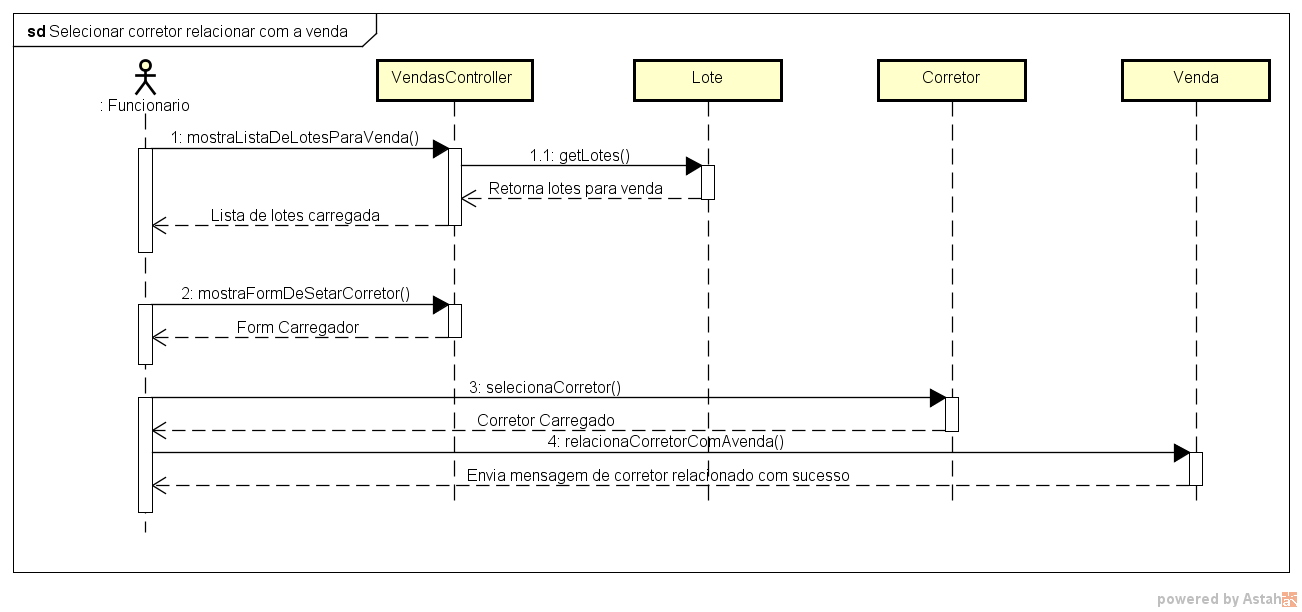
Figura 14 – Diagrama de Classe



Fonte: **PRÓPRIA, 2016**

### Selecionar corretor que efetuou a venda

Figura 15 – Diagrama de Sequência



Fonte: **PRÓPRIA, 2016**

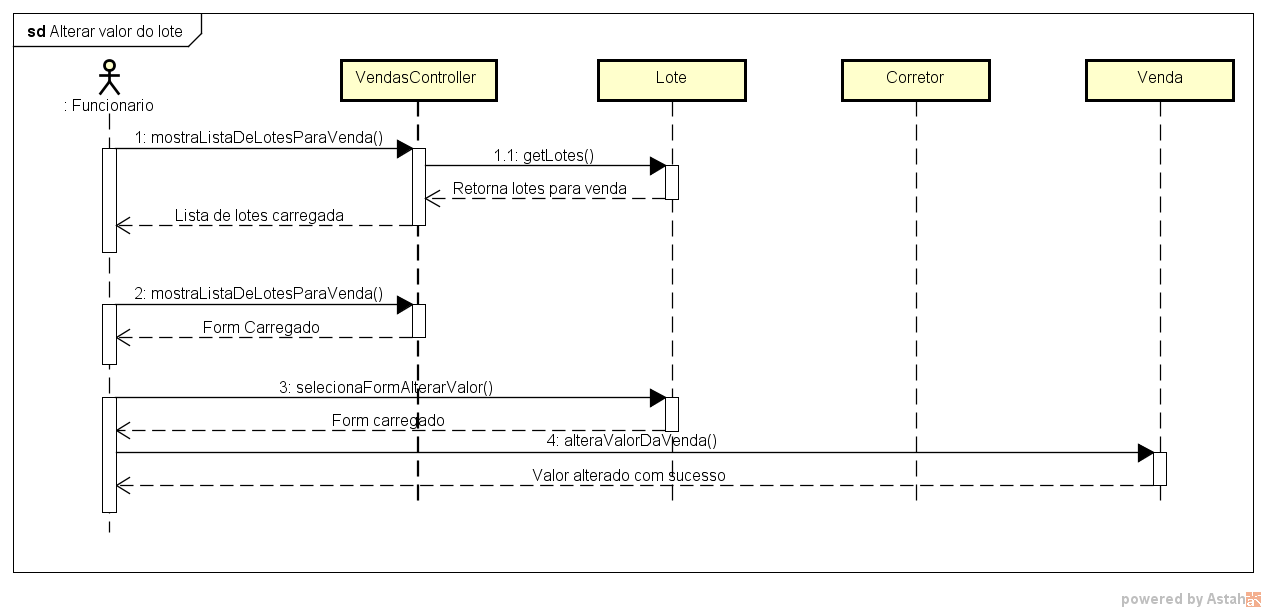
Figura 16 – Diagrama de Classe



Fonte: **PRÓPRIA, 2016**

### Alterar valor do lote

Figura 17 – Diagrama de Sequência



Fonte: **PRÓPRIA, 2016**

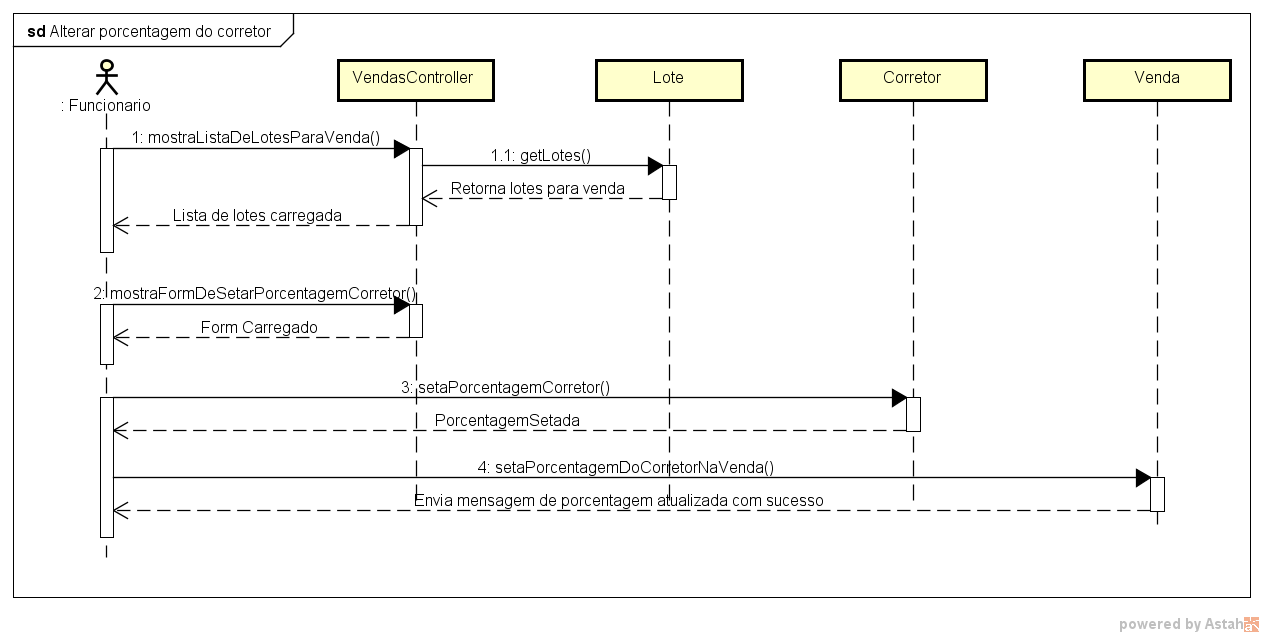
Figura 18 – Diagrama de Classe



Fonte: **PRÓPRIA, 2016**

### Alterar porcentagem do corretor

Figura 19 – Diagrama de Sequência



Fonte: **PRÓPRIA, 2016**

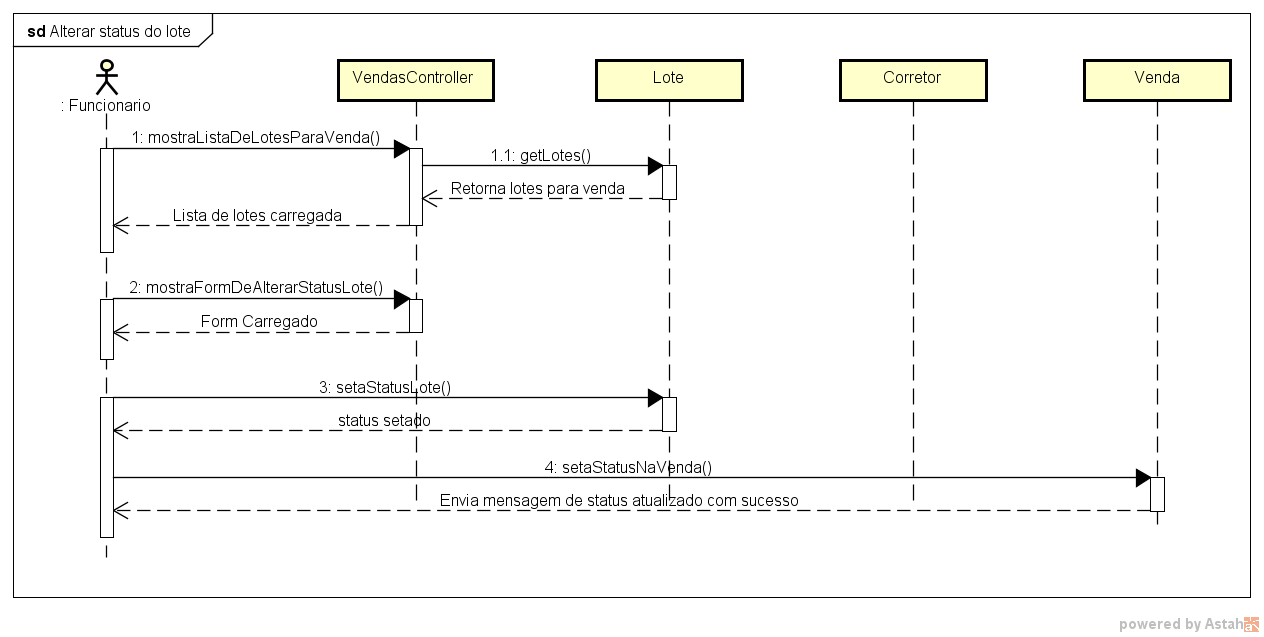
Figura 20 – Diagrama de Classe



Fonte: **PRÓPRIA, 2016**

### Alterar status do lote

Figura 21 – Diagrama de Sequência



Fonte: **PRÓPRIA, 2016**

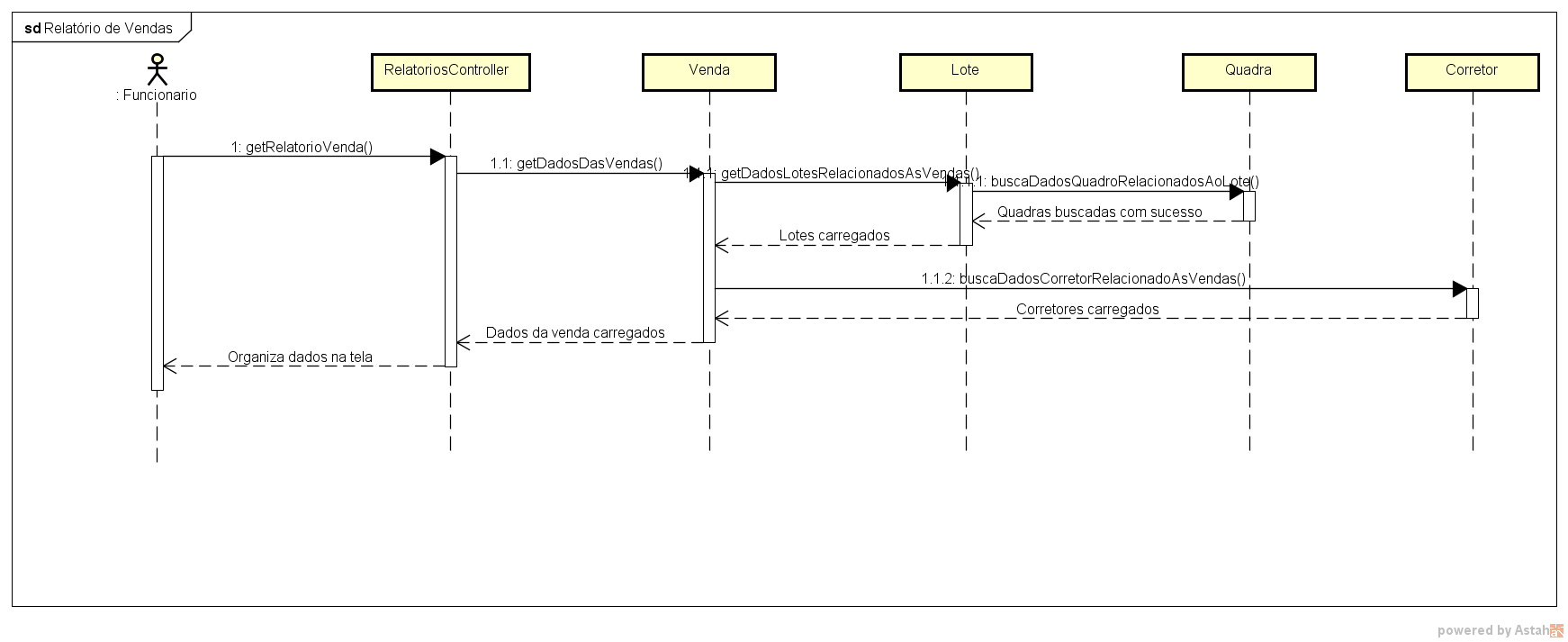
Figura 22 – Diagrama de Classe



Fonte: **PRÓPRIA, 2016**

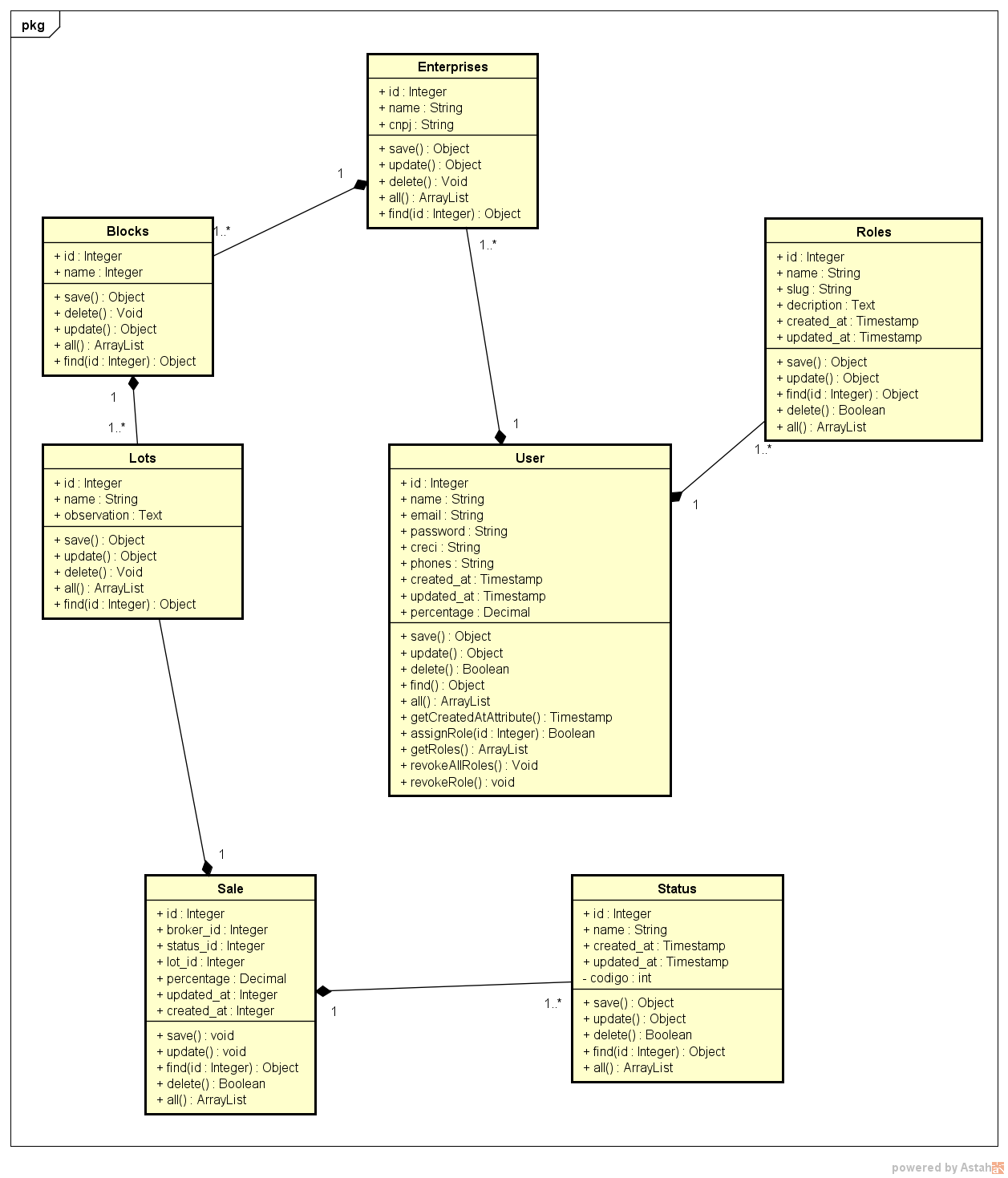
### Relatório de Vendas

Figura 23 – Diagrama de Sequência



Fonte: **PRÓPRIA, 2016**

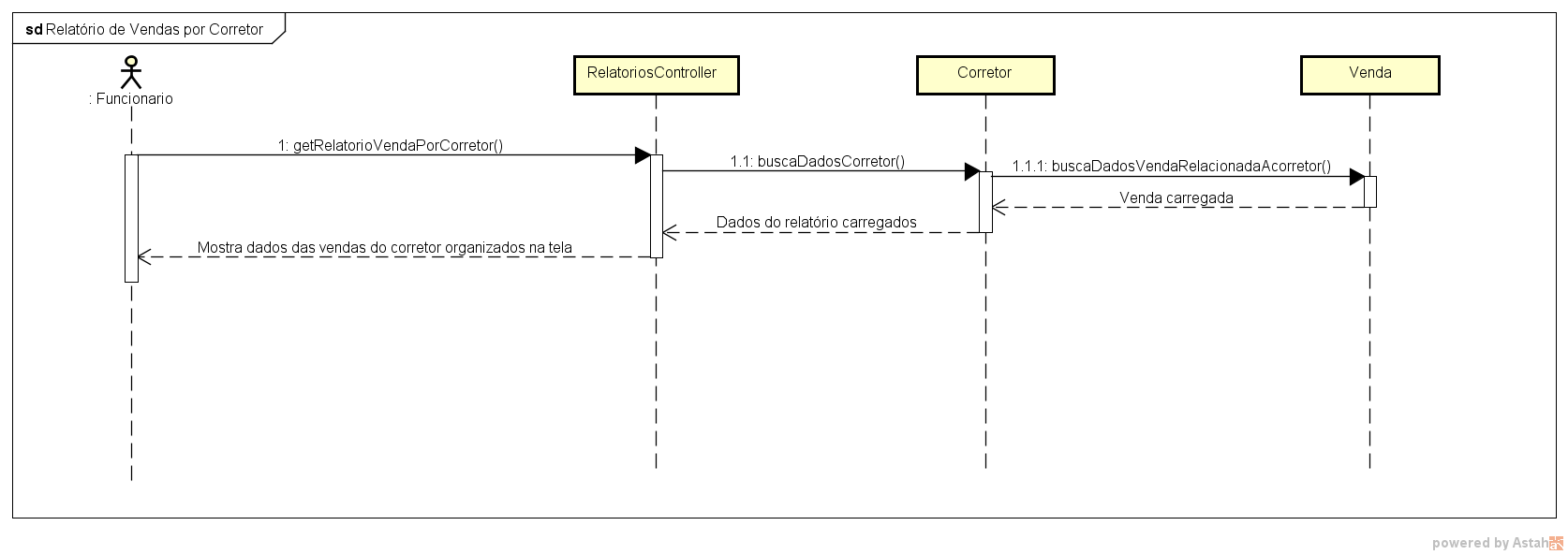
Figura 24 – Diagrama de Classe



Fonte: **PRÓPRIA, 2016**

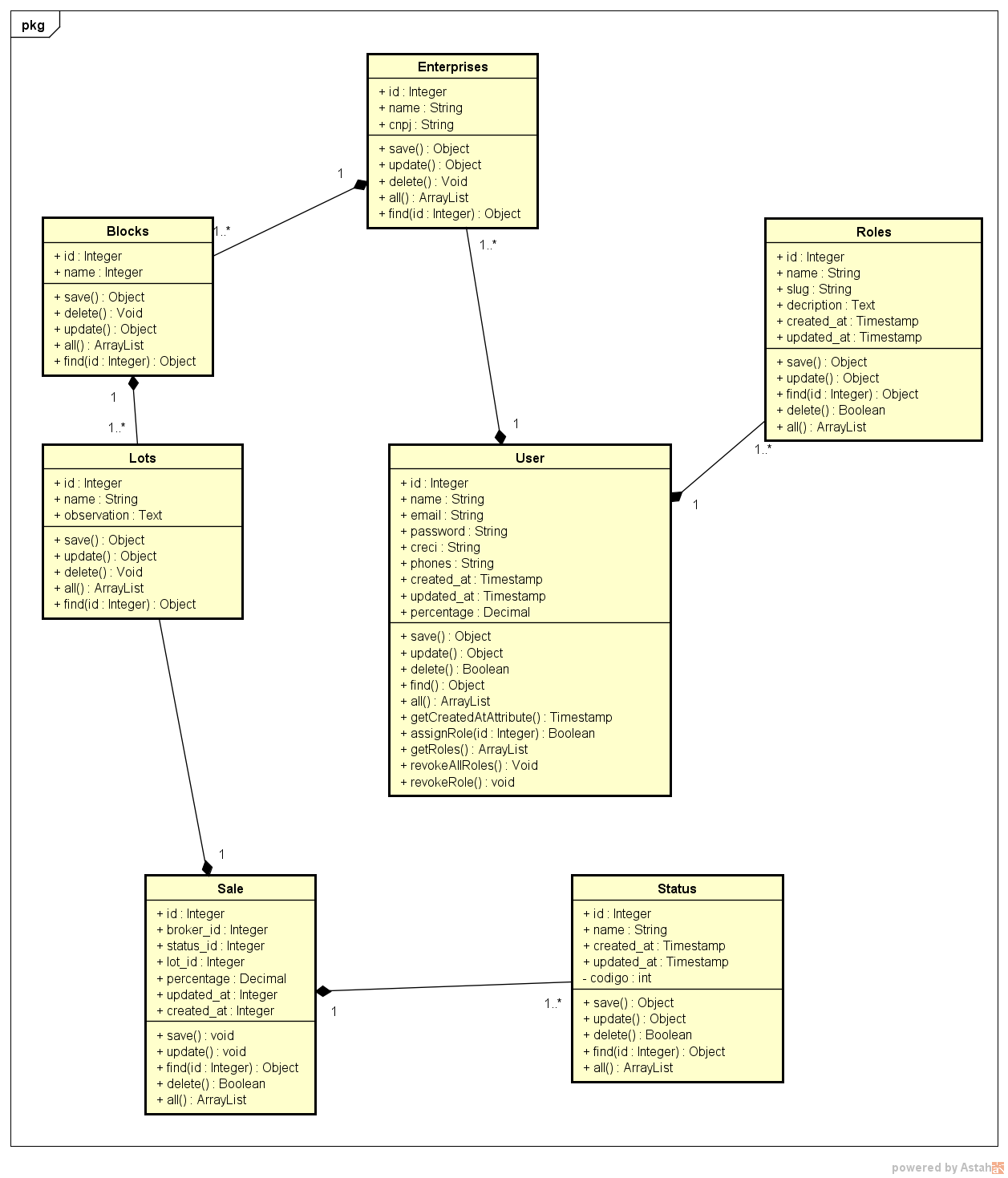
### Relatório de Vendas por Corretor

Figura 25 – Diagrama de Sequência



Fonte: **PRÓPRIA, 2016**

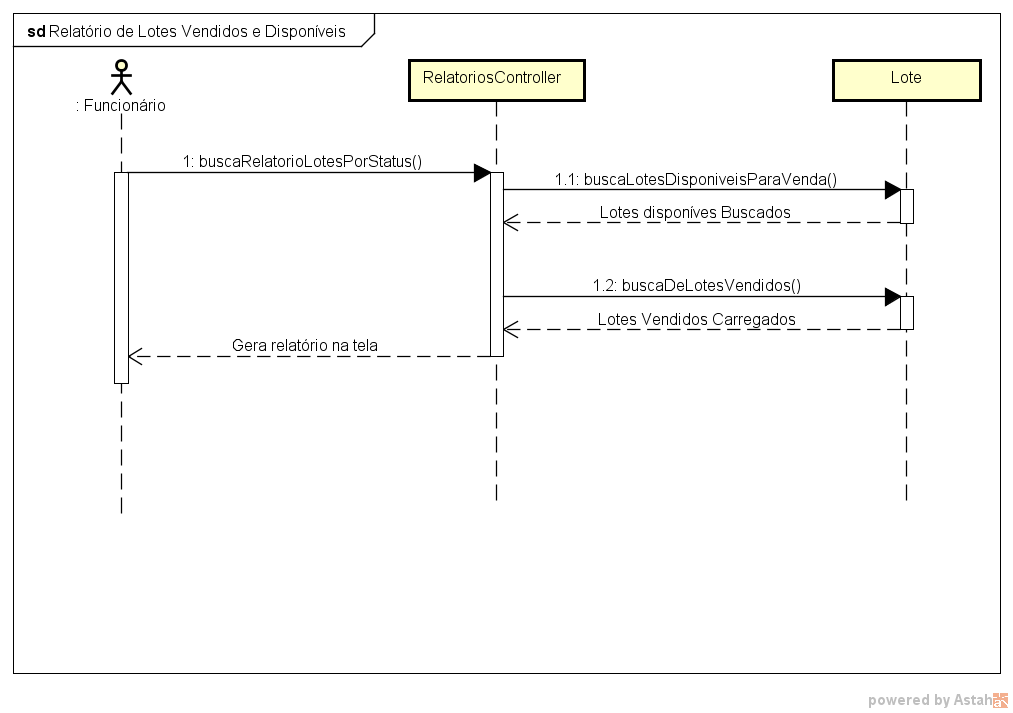
Figura 26 – Diagrama de Classe



Fonte: **PRÓPRIA, 2016**

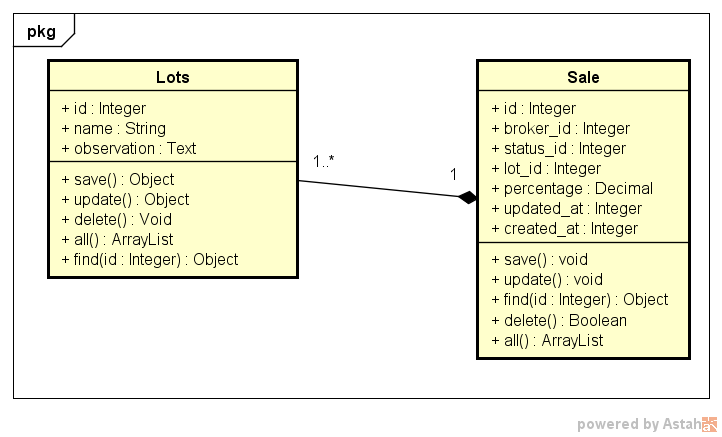
### Relatório de Lotes Vendidos e Disponíveis

Figura 27 – Diagrama de Sequência



Fonte: **PRÓPRIA, 2016**

Figura 28 – Diagrama de Classe



Fonte: **PRÓPRIA, 2016**

### Construir por Funcionalidade

Essa é a fase onde é iniciada a codificação das funcionalidades bem como a entrega das mesmas. É listada a seguir, no **Quadro 4,** as interfaces finais do projeto, bem como as funcionalidades entregue por cada tela e as imagens de cada tela.

**Quadro 4 – Lista de Funcionalidades por Interface**

|  |  |
| --- | --- |
| **Interface** | **Funcionalidades** |
| Interface de *Login* | - *Login* no Sistema |
| Interface de Cadastro de Cliente | - Cadastro de Usuário Administrador |
| Funcionários | - Lista de Funcionários  - Exclusão de Funcionários  - Log |
| Salvar Funcionário | - Cadastro de Funcionário |
| Corretores | - Listagem de Corretores  - Exclusão de Corretores  - *Log* |
| Empreendimentos | - Listagem de Empreendimentos  - Exclusão de Empreendimentos |
| Quadras/Lotes | - Listagem de Quadras  - Exclusão de Quadra  - Listagem de Lotes  - Exclusão de Lotes  - Cadastro de Quadra  - Cadastro de Lote |
| Vendas | - Selecionar Corretor que efetuou a venda  - Alterar valor do lote  - Alterar porcentagem do corretor  - Alterar *Status* do Lote |

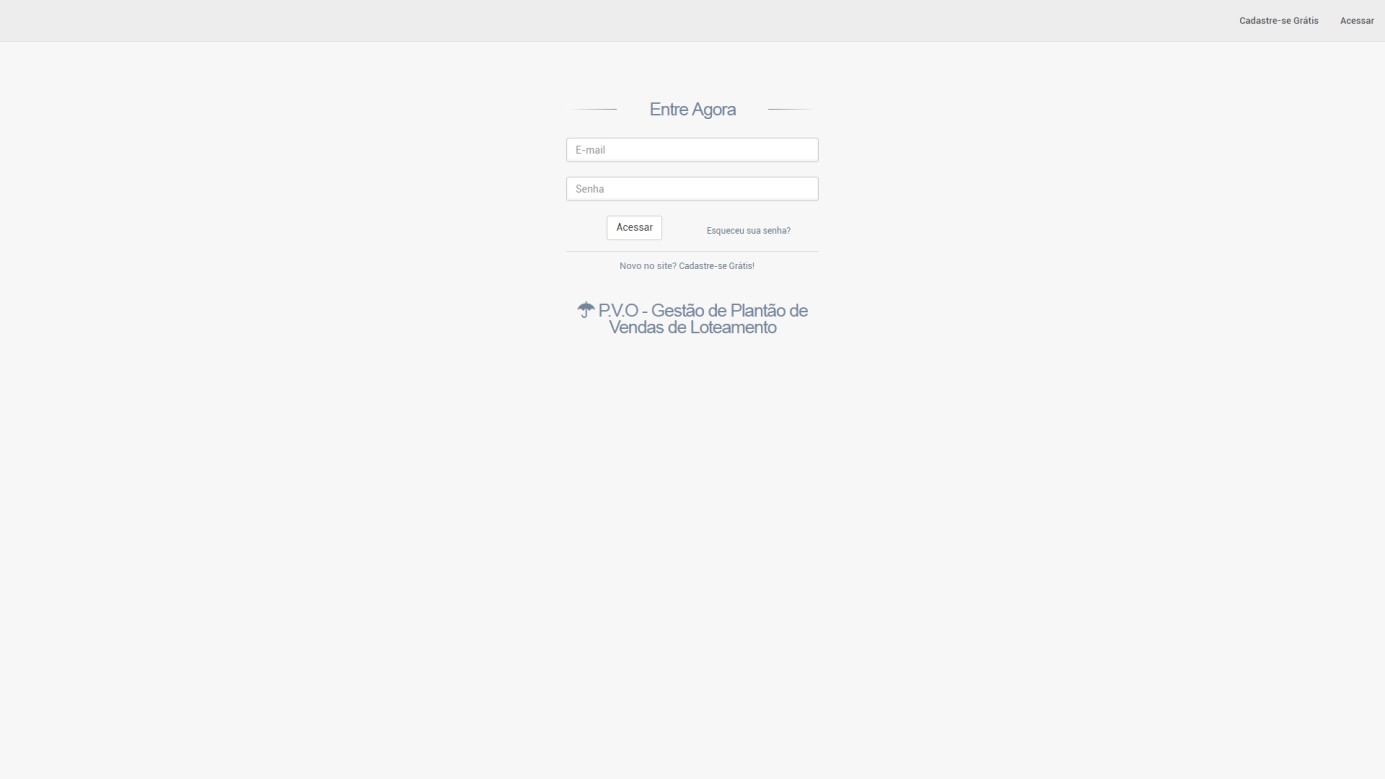
Fonte: **PRÓPRIA, 2016**.

### Interface de Login

A interface de *login,* como apresentado na FIG. 29 é a representação final da funcionalidade “Login no sistema”. O Fluxo principal da funcionalidade fazer login envolve as seguintes tarefas:

1. Os usuários iniciam o processo de *login* abrindo a página de *login* que está disponível na url https://pacific-thicket-57678.herokuapp.com/[https://pacific-thicket-57678.herokuapp.com/](https://pacific-thicket-57678.herokuapp.com)
2. Os usuários informando todos os dados solicitados para fazer o *login*.
3. Os usuários fazem *login*

Figura 29 – Tela de Login



Fonte: **PRÓPRIA, 2016**.

O Quadro 5 demonstra os campos da Tela “Login”. Os campos devem ser preenchidos de acordo, para que o usuário tenha acesso ao sistema.

**Quadro 5 – Lista de campos Tela de *Login***

|  |  |  |  |
| --- | --- | --- | --- |
| **Nome** | **Descrição** | **Tipo** | **Restrições** |
| *E-mail* | *E-mail* cadastrado no sistema para *login* | Texto | Obrigatório |
| Senha | Senha do cliente cadastrado | Texto | Obrigatório |

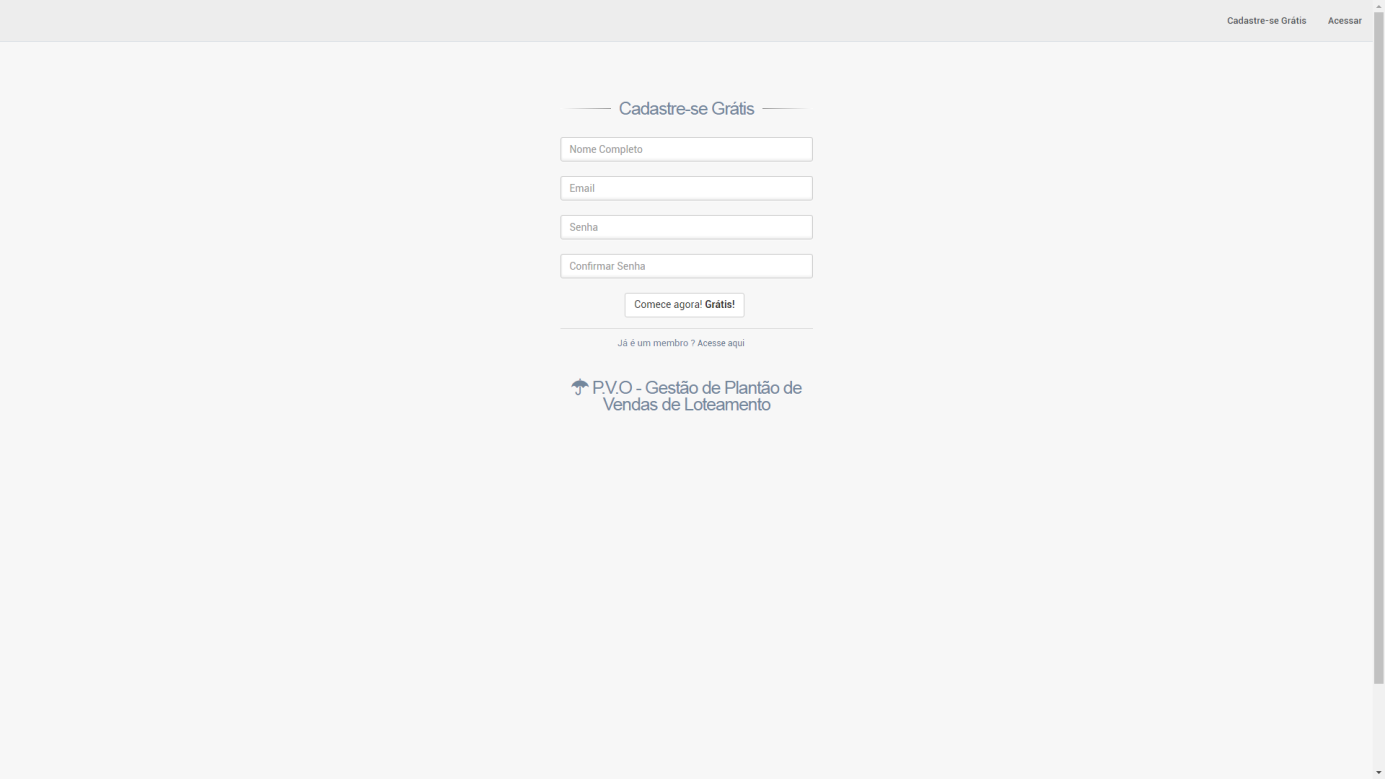
Fonte: **PRÓPRIA, 2016**.

### Interface de Cadastro de Cliente

A interface de Cadastro do Cliente (FIG. 30) tem a função de cadastrar o usuário administrador do sistema. O Fluxo Principal desta funcionalidade permite que:

1. Os usuários entram na url https://pacific-thicket-57678.herokuapp.com/[https://pacific-thicket-57678.herokuapp.com/](https://pacific-thicket-57678.herokuapp.com)e clicam no *link* “Cadastre-se Grátis”.
2. Os usuários entram com todos os campos necessários para seu cadastro.
3. Os usuários se cadastram.

Figura 30 – Tela de Cadastro de Cliente



Fonte: **PRÓPRIA, 2016**.

**Quadro 6 – Lista de campos Tela de Login**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nome** | **Descrição** | **Tipo** | **Restrições** |
| Nome Completo | Nome completo do cliente que está efetuando cadastro | Texto | Obrigatório |
| E-mail | E-mail do cliente. Esse e-mail será o *login* do usuário | Texto | Obrigatório |
| Senha | Senha do cliente | Texto | Obrigatório |
| Confirmar Senha | Confirmação da senha do cliente | Texto | Obrigatório  Ser igual ao campo senha |

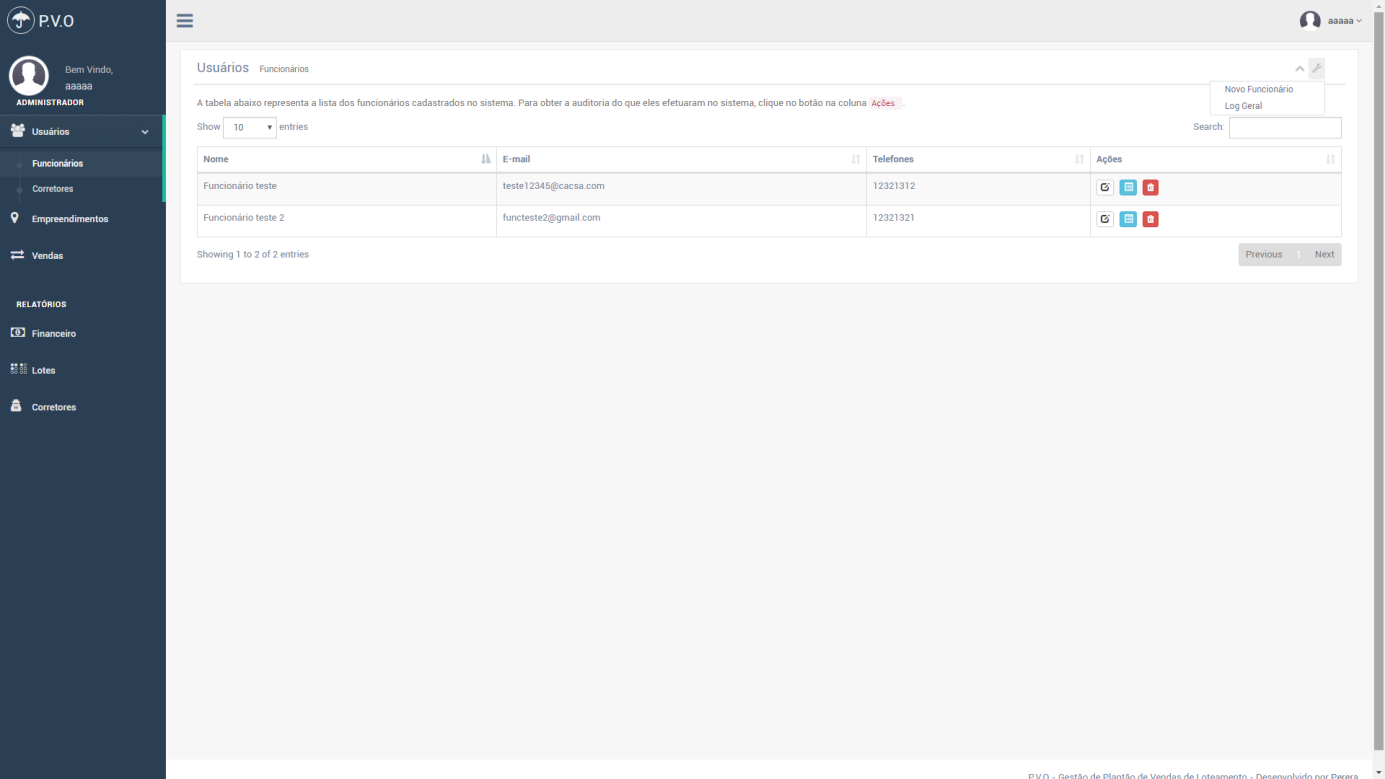
Fonte: **PRÓPRIA, 2016**.

### Interface de Funcionários

A interface de Funcionários tem a função de listar os usuários que são funcionários de um determinado cliente e os excluir. O Fluxo principal das funcionalidades são:

1. O usuário clica no *link* “Usuários>funcionários”.
2. Aparece a Lista de Funcionários cadastrados
3. O usuário clica em “excluir”, que é o ícone de uma lixeira, caso queira excluir.

Figura 31 – Tela de Funcionários



Fonte: **PRÓPRIA, 2016**.

**Quadro 7 – Comandos da Tela de Funcionários**

|  |  |  |
| --- | --- | --- |
| **Nome** | **Ação** | **Restrição** |
| Editar Funcionário | Entra no formulário para edição dos dados do funcionário | - |
| Excluir funcionário | Apaga um funcionário do Sistema | Somente Usuário Administrador |
| Novo Funcionário | Entra na página de Cadastrar um funcionário | - |
| *Log* | Visualiza *Log* do funcionário | - |

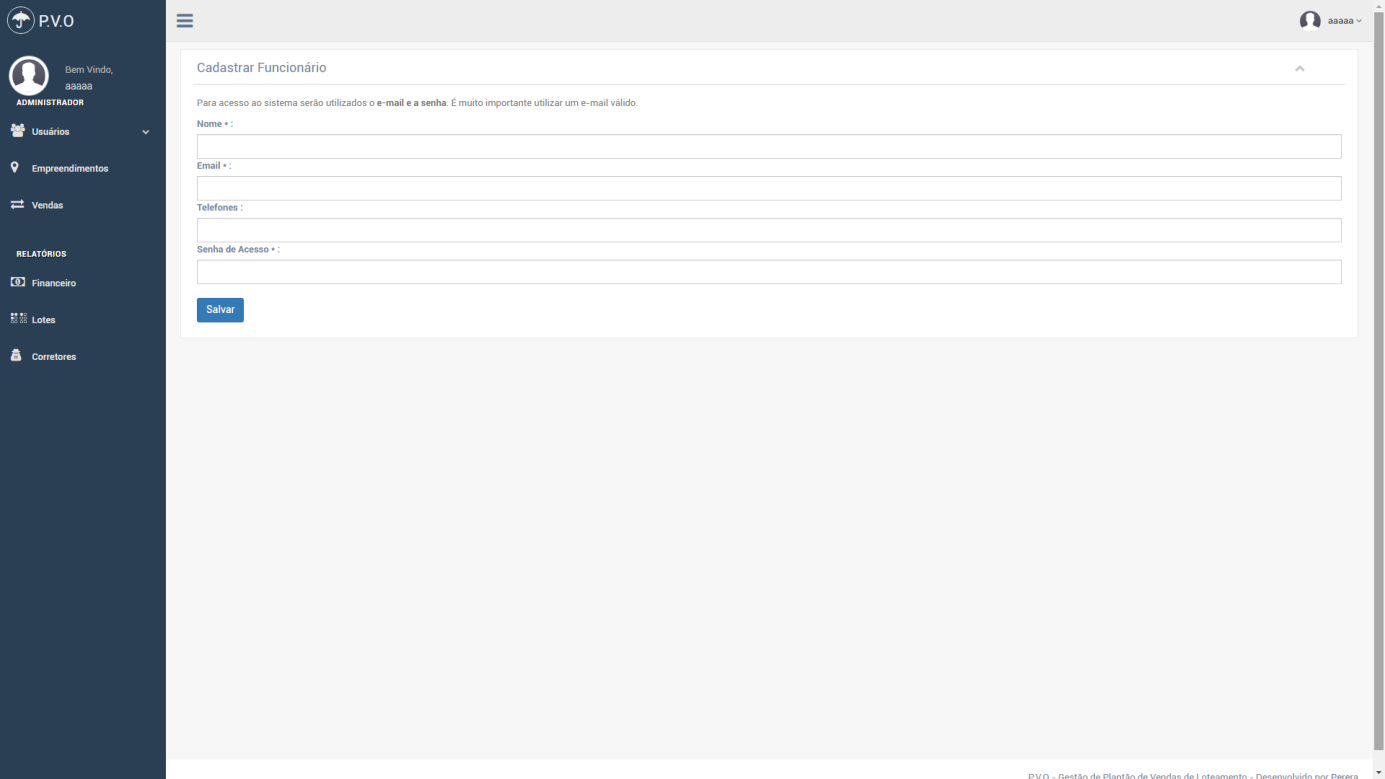
Fonte: **PRÓPRIA, 2016**.

### Interface de Salvar Funcionário

A interface de salvar funcionário irá gerar o formulário para cadastro de um novo funcionário no sistema. O Fluxo principal das funcionalidades é:

1. O usuário clica no *link* “Usuários>funcionários”.
2. O usuário clica no ícone de uma ferramenta no canto superior direito e após em “Novo funcionário”
3. O usuário cadastra um novo funcionário.

Figura 32 – Tela de Salvar Funcionário



Fonte: **PRÓPRIA, 2016**.

**Quadro 8 – Lista de campos da Tela de Funcionários**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nome** | **Descrição** | **Tipo** | **Restrições** |
| Nome | Nome completo do funcionário que está efetuando cadastro | Texto | Obrigatório |
| E-mail | E-mail do funcionário. Esse e-mail será o *login* do usuário | Texto | Obrigatório |
| Telefones | Telefone do funcionário | Texto | - |
| Senha de Acesso | Senha para *login* no sistema | Texto | Obrigatório |

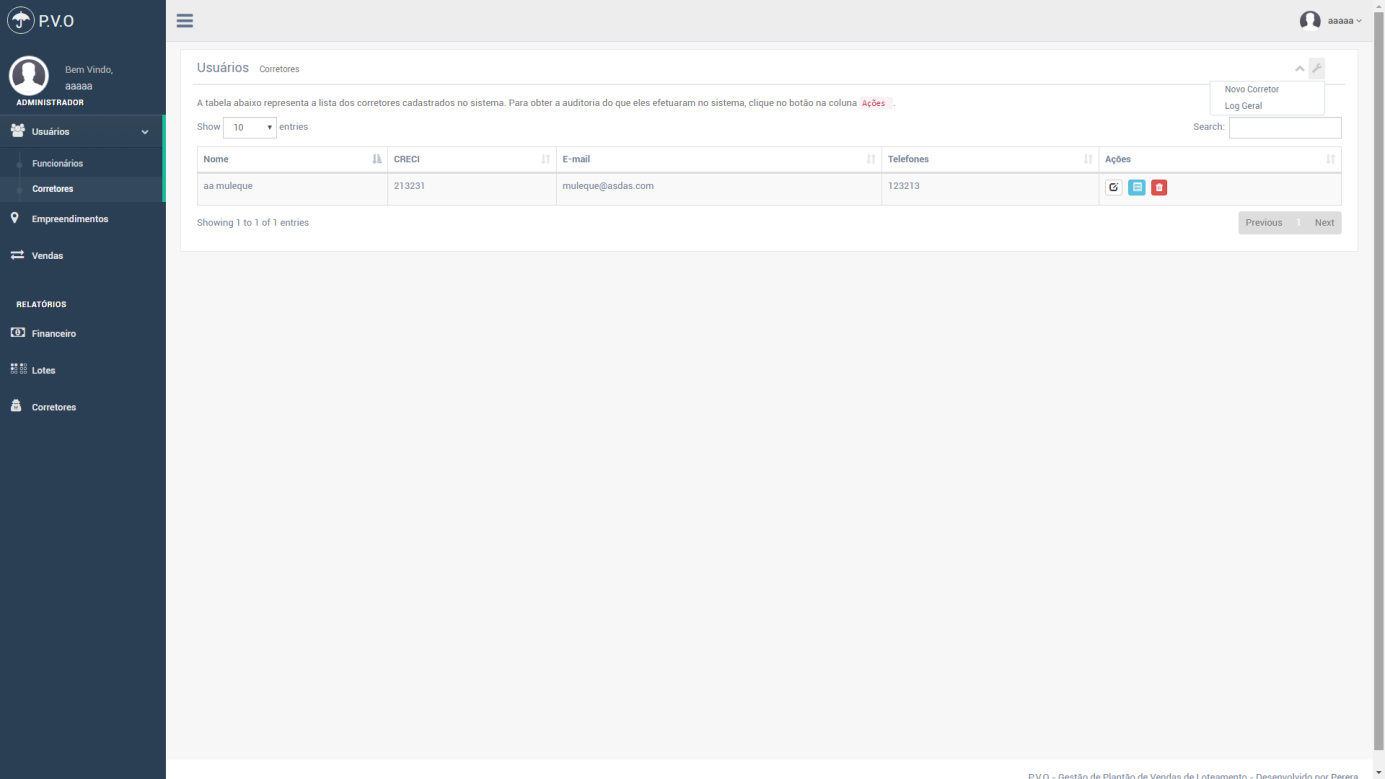
Fonte: **PRÓPRIA, 2016**.

### Interface de Corretores

A interface de Corretores tem a função de listar os usuários que são corretores de um determinado cliente e os excluir. O Fluxo principal das funcionalidades são:

1. O usuário clica no *link* “Usuários>corretores”.
2. Aparece a Lista de Corretores cadastrados
3. O usuário clica em “excluir”, que é o ícone de uma lixeira, caso queira excluir.

Figura 33 – Tela de Corretores



Fonte: **PRÓPRIA, 2016**.

**Quadro 9 – Comandos da Tela de Corretores**

|  |  |  |
| --- | --- | --- |
| **Nome** | **Ação** | **Restrição** |
| Editar Corretor | Entra no formulário para edição dos dados do corretor | - |
| Excluir Corretor | Apaga um corretor do Sistema | Somente Usuário Administrador |
| Novo Corretor | Entra na página de Cadastrar um corretor | - |
| Log | Visualiza Log do corretor | - |

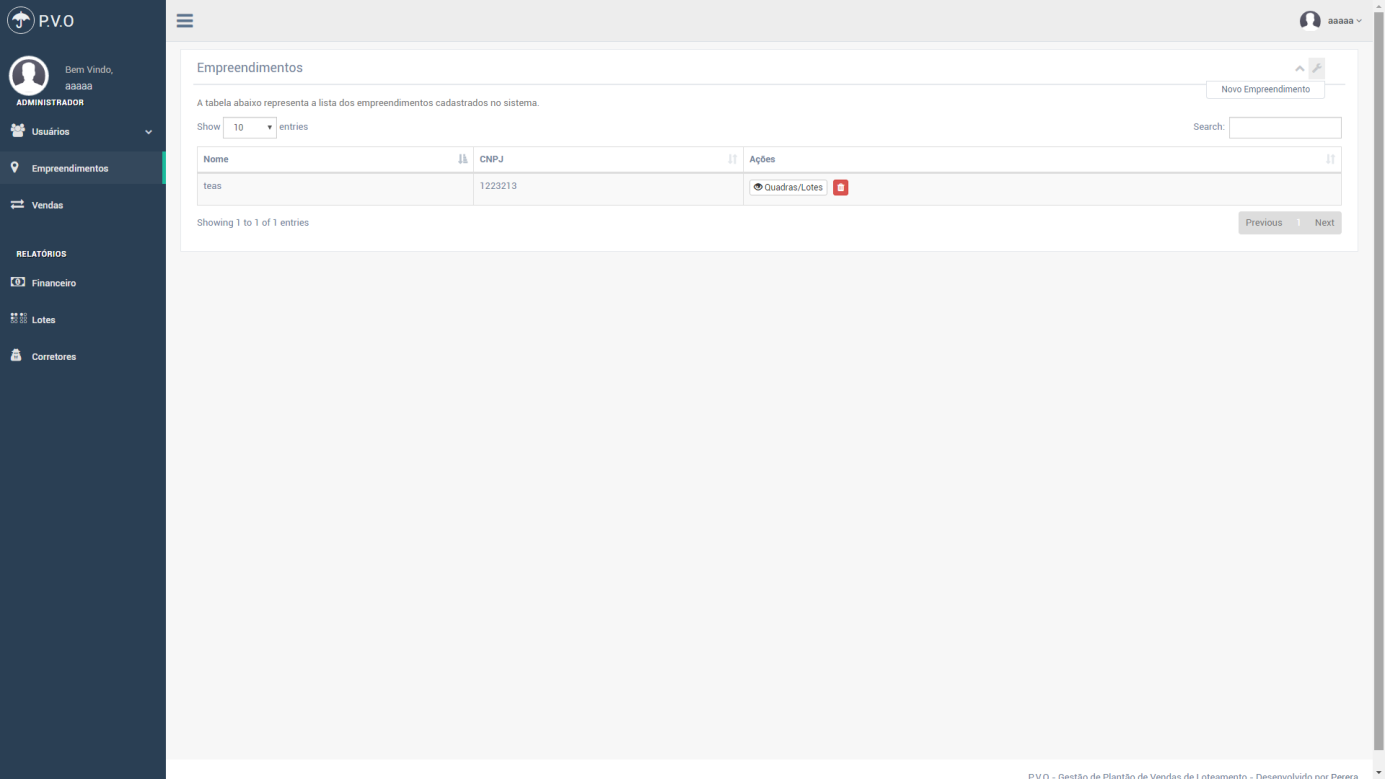
Fonte: **PRÓPRIA, 2016**.

### Interface de Empreendimentos

A interface de Empreendimentos tem a função de listar os empreendimentos que são gerenciados no sistema. O Fluxo principal das funcionalidades são:

1. O usuário clica na aba “Empreendimentos”
2. Aparece a lista de empreendimentos cadastrados
3. O usuário clica em excluir para excluir ou novo empreendimento para cadastro.

Figura 34 – Tela de Empreendimentos



Fonte: **PRÓPRIA, 2016**.

**Quadro 10 – Comandos da Tela de Corretores**

|  |  |  |
| --- | --- | --- |
| **Nome** | **Ação** | **Restrição** |
| Quadras/Lotes | Entra na tela de gestão de quadras e lotes relacionadas ao empreendimento | - |
| Excluir Empreendimento | Apaga um empreendimento do sistema | Somente Usuário Administrador |
| Novo Empreendimento | Entra no formulário de cadastro de empreendimento | - |

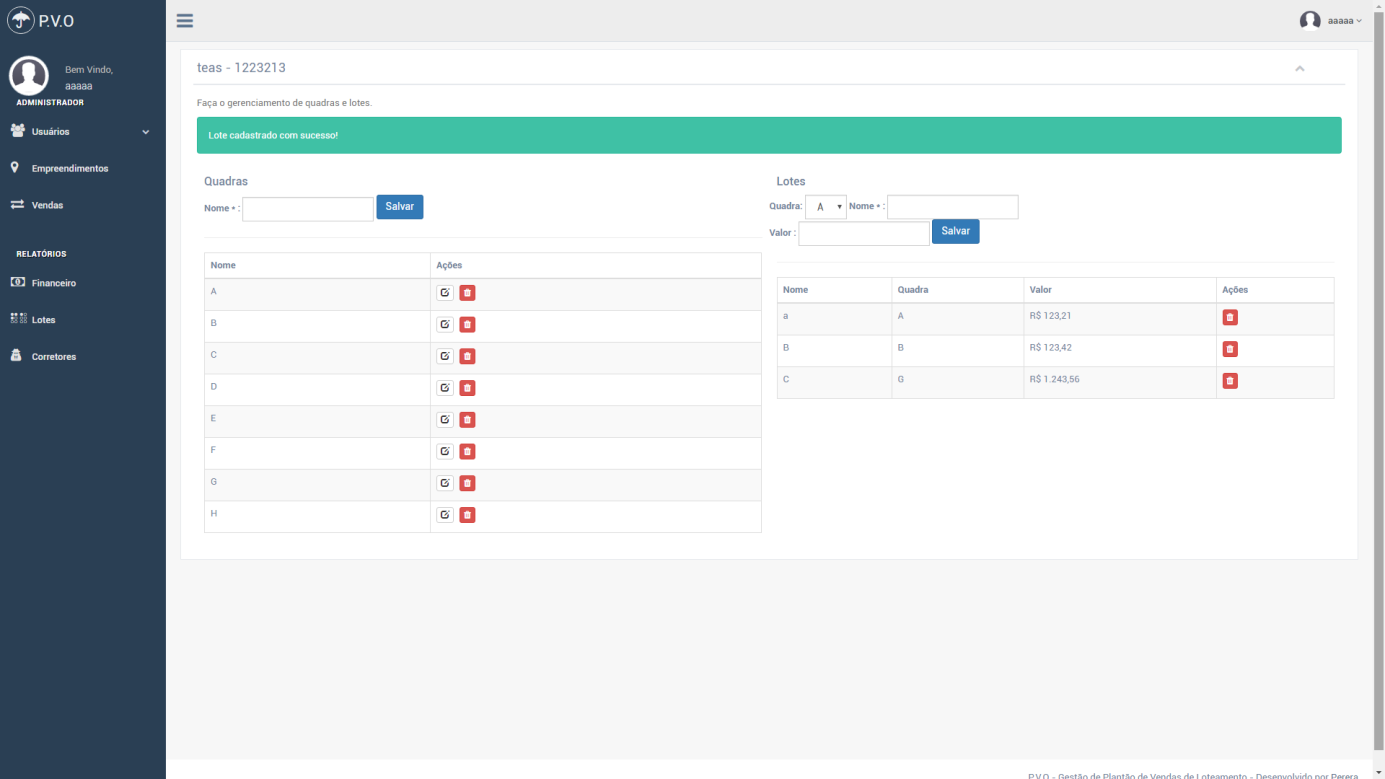
Fonte: **PRÓPRIA, 2016**.

### Quadras/lotes

A interface de quadras e lotes entrega as funcionalidades de cadastrar, excluir e listar tanto as quadras quanto os lotes. Segue lista do fluxo principal:

1. O usuário entra na aba “Empreendimentos” e logo após em “Quadras/lotes”
2. Irá aparecer a lista de quadras e de lotes
3. O usuário excluir quadra/lote ou cadastra quadra/lote.

Figura 35 – Tela de Quadras/Lotes



Fonte: **PRÓPRIA, 2016**.

**Quadro 11 – Comandos da Tela de Quadras/Lotes**

|  |  |  |
| --- | --- | --- |
| **Nome** | **Ação** | **Restrição** |
| Salvar Quadra | Cadastrar uma nova quadra no empreendimento em questão. | - |
| Excluir Quadra | Apaga uma quadra do empreendimento em questão | Somente Usuário Administrador |
| Editar Quadra | Edita o nome de uma quadra | - |
| Salvar Lote | Cadastrar um novo lote relacionado a uma quadra | - |
| Excluir Lote | Exclui um lote do sistema | - |

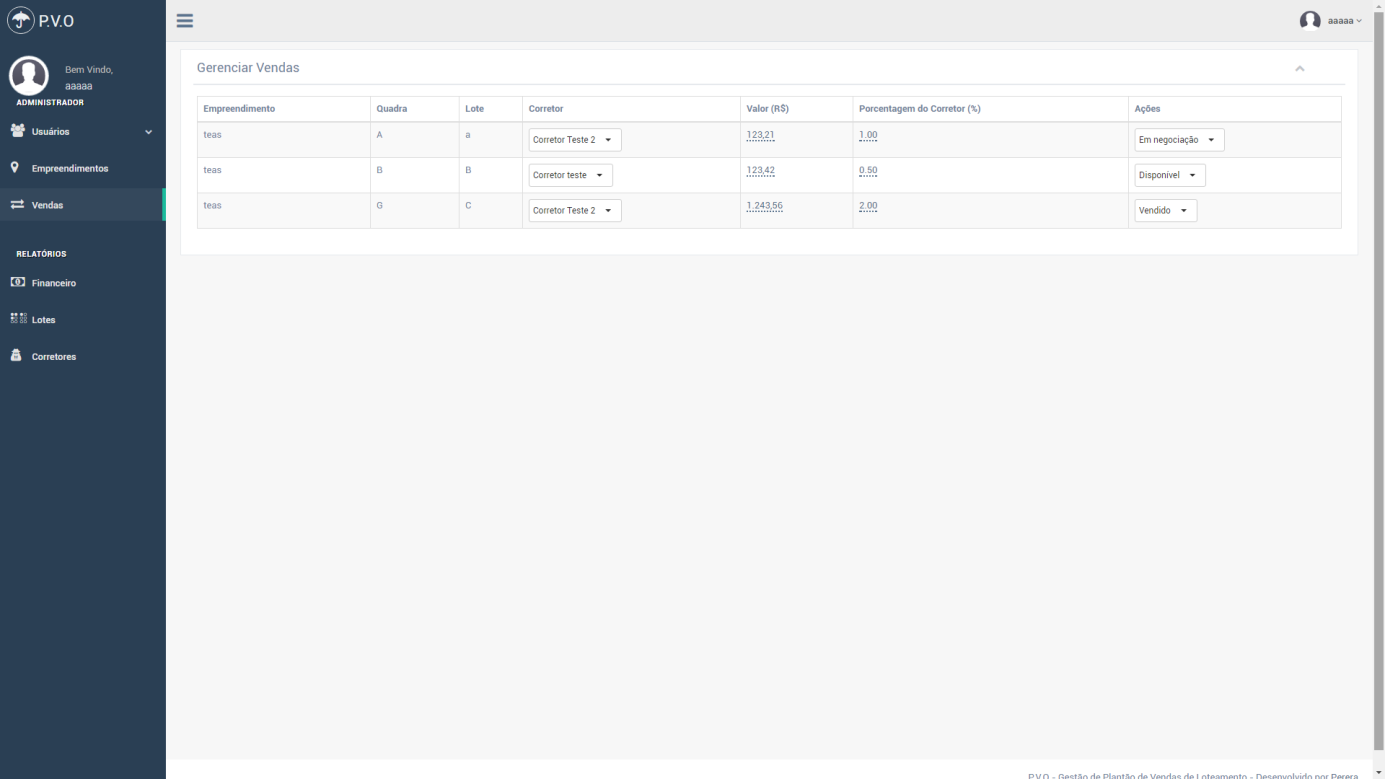
Fonte: **PRÓPRIA, 2016**.

### Vendas

A tela de Vendas é a responsável pela gestão da venda dos lotes, incluindo valor do lote, porcentagem para corretor e status atualizado do lote. O fluxo principal é:

1. Usuário acessa a aba “Vendas”
2. Aparece a lista de lotes do sistema
3. O usuário faz a gestão dos lotes de acordo.

Figura 36 – Tela de Vendas



Fonte: **PRÓPRIA, 2016**.

**Quadro 12 – Comandos da Tela de Vendas**

|  |  |  |
| --- | --- | --- |
| **Nome** | **Ação** | **Restrição** |
| Selecionar Corretor que efetuou a venda | Relaciona a venda ao corretor que vendeu o lote | - |
| Alterar valor do lote | Seta o valor do lote vendido em questão. | - |
| Alterar porcentagem do corretor | Seta o valor da porcentagem que o corretor ganhará caso venda o lote. | - |
| Alterar Status do Lote | Seta o status do lote. | - |

Fonte: **PRÓPRIA, 2016**.

### Usuários e sistemas externos

Os usuários são mostrados no Quadro 13.

**Quadro 13 – Tipos de Usuário**

|  |  |
| --- | --- |
| **Ator** | **Definição** |
| Administrador (Cliente) | É responsável por gerenciar todo o sistema. Suas principais funções são o acompanhamento de venda e funcionários através dos relatórios. |
| Funcionário | É quem faz a gerência geral do sistema. Cadastros, corretores, valores, porcentagens, etc. |
| Corretor | Corretor é o usuário que vende o lote e tem acesso aos dados específicos da sua venda. |

# RESULTADOS OBTIDOS

O sistema *web* de plantão de vendas de loteamento é uma ferramenta de apoio a gestão no momento da venda de lotes pelos corretores, tendo a função de informar caso o lote tenha sido vendido e o cálculo de porcentagem exata do valor do lote que irá para o respectivo corretor. Além disso, é possível ranquear os corretores pelo valor arrecadado sabendo qual o corretor obteve maior lucro para a empresa.

O sistema **gera** a padronização de processos bem como **regula** e **gerencia** as ações de funcionários e corretores no período de vendas dos lotes. O sistema criou transparência de resultados para os corretores bem como para o sócio proprietário que também pode auditar as ações do funcionário que é o responsável por cadastrar os lotes e respectivos valores, bem como alterar os *status* da venda do lote.

O sistema foi desenvolvido com o *framework* Laravel no *Serverside* e utilizando o SGBD **MySQL** para gerenciamento de dados, bem como as tecnologias css, javascript e HTML para o *front-end* do sistema.

Anteriormente a realização do processo ocorria manualmente, sem a possibilidade de auditar e havia inúmeros problemas em relação a porcentagem correta para cada corretor devido a baixa transparência de informações. A informação do *status* atual de um lote era efetuado no “boca a boca” o que ocasionava a venda de um mesmo lote por mais de um corretor para mais de um cliente.

O sistema possibilitou a capacidade do corretor ter em mãos rapidamente o *status* de um lote diminuindo o risco de erros no instante da venda de um lote, otimizando a possibilidade de venda dos lotes em geral.

O sistema possibilitou o cadastro de usuários com diferentes tipos de permissões de acesso. Existem três tipos de perfis de usuários: o administrador (Sócio Proprietário do Loteamento), o corretor (agente autônomo na venda dos lotes) e o funcionário da empresa de loteamento. O administrador tem acesso a todas as áreas do sistema, podendo também auditar a ação de todos os usuários no sistema e adicionar qualquer que seja o tipo de usuário.

O corretor tem acesso ao seu relatório financeiro atualizado e ao *status* e valores de cada lote, conferindo transparência e agilidade ao processo da venda e o funcionário tem acesso ao cadastramento de empreendimentos, quadras, lotes, corretores e relatórios de lotes vagos/utilizados, financeiro geral e *ranking* de corretores.

O sistema possibilita também a exportação em formato “pdf” do relatório geral de vendas dos lotes e do relatório por corretor com o intuito de ser documento comprobatório de ação do processo de venda.

O sistema possibilita a visualização de todos os dados necessários no momento de um plantão de vendas de forma agradável e com boa usabilidade.

# CONSIDERAÇÕES FINAIS

O trabalho consiste no desenvolvimento de um sistema *web* de gerenciamento plantão de vendas para a empresa GrandMinas. O sistema apresenta área de relatório para o corretor do loteamento, e essa área tem como função resolver o problema de transparência em relação aos valores que serão recebidos pelos corretores, o qual tem todos os detalhes da venda bem como o valor correto da porcentagem do corretor.

Foi demonstrado também que houve a necessidade do acesso ao sistema de qualquer lugar do mundo e relacionado a isso foi tomada a decisão de criar um sistema *web* hospedado no **heroku**, pois existe a possibilidade de acesso em qualquer lugar do mundo através da internet, sanando o problema do acesso.

Outro fator relevante é a centralização e clareza de informações em um só lugar. Está criada no sistema a tela de vendas, onde, todas as informações relativas à venda de um determinado lote estão nessa tela, sendo assim, os usuários corretor, funcionário e administrador tem acesso à informação em um local único o que facilita a comunicação e o trabalho dos usuários.

Nota-se também a facilidade de saber o valor único e total das vendas de lotes e quanto dessas vendas serão revertidas em bonificações para os corretores, além de contar com um sistema de *ranking* de corretores onde pode medir o corretor que vendeu uma maior quantidade de lotes.

É recomendado para elaboração de trabalhos futuros: a integração com o sistema de gerenciamento financeiro de toda a empresa de loteamento e a integração com *site* de vendas de lotes on-line. Dessa maneira a gestão do loteamento será integrada com o sistema contábil, gerando os contratos para o cliente da GrandMinas e consequente geração e envio dos boletos de cobrança do lote, automatizando toda a cadeia de processos de vendas e cobrança da empresa GrandMinas. Com a integração com *site* de venda de loteamento *on-line* há possibilidade de um novo canal de vendas, com a atualização de status do lote automática independente do canal utilizado para a venda.

Levando em consideração o que foi apresentado anteriormente, pode-se então concluir que, o trabalho foi executado com êxito cumprindo com os objetivos gerais e específicos.

# REFERÊNCIAS

1. ADRIEL, Wendell; **Introdução ao Laravel Framework PHP.** 2015 Disponível em: http://www.devmedia.com.br/introducao-ao-laravel-framework-php/33173Acessado em: 20 de set. 2016
2. **ASTAH\* BASIC OPERATION GUIDE, 2009.** Disponível em: <http://astah.net/resources/documents/astah-basic-operation.pdf> Acesso em: 25 de agosto 2016.
3. BOOCH, Grady; RUMBAUGH, James; JACOBOSN, Ivar; **UML : Guia do Usuário**. Elsevier, 2005.
4. CARRIL, Marly; **HTML – Passo a passo 2012.**
5. **DOCUMENTAÇÃO OFICIAL JQUERY.** Disponível em: <https://jquery.com/> Acesso em: 20 de setembro de 2016
6. JOBSTRAIBIZER, Flávia; **CRIAÇÃO DE SITES COM CSS.** Editora Digerati, 2009

LAUDON, K.C.; LAUDON, J.P. **Sistemas de Informações Gerencias**. 7. ed. São Paulo: Person Prentice Hall, 2007.

LAUDON, K.C.; LAUDON, J.P. **Sistemas de Informações Gerencias**. 9. ed. São Paulo: Person Prentice Hall, 2009.

LOBO, E. J. R. **Curso de Engenharia de Software.** Editora Digerati Books: São Paulo, 2008.

LOBO, Edson; **Curso Prático de MySQL.** Digerati Books, 2008.

MANSUR, Ricardo. **Governança de TI: Metodologias, Frameworks e Melhores Práticas.** Brasport Livros e Multimídia Ltda: Rio de Janeiro, 2007.

MASSARI, Vitor L. **Agile Scrum Master no Gerenciamento Avançado de Projetos.** Brasport Livros e Multímidia Ltda: Rio de Janeiro, 2016.

MELO, Ana Cristina; **Desenvolvendo aplicações com UML 2.2 – Do conceitual a implementação.** Brasport Livros e Multimídia LTDA: Rio de Janeiro, 2010 – 3ª ed.

MUTO, Claudio Adonai; **PHP & MYSQL – GUIA INTRODUTÓRIO,** 3ª Ed. 2006 Brasport

NIELSEN, Jakob; LORANGER, Hoa; **Usabilidade na web – Projetando Websites com Qualidade.** Elsevier Editora Ltda: Rio de Janeiro, 2007.

PRESCOTT, Preston; **Programação em Javascript.** Babelcube

PRESSMAN, R. S. **Engenharia de Software: Uma abordagem profissional.** 7. ed. Editora Bookman.

PRESSMAN, R. S.; MAXIM, B. R. **Engenharia de Software: Uma abordagem profissional.** 8. ed. Editora Bookman.

RAMAKRISHNAN, Raghu; GEHRKE, Johannes; **SISTEMAS DE GERENCIAMENTO DE BANCO DE DADOS.** Editora Bookman, Mc Graw Hill, 2008

REZENDE, Denis Alcides. **Engenharia de Software e Sistemas de Informação.** 3ª Edição – Editora Brasport: Rio de Janeiro, 2005.

SAMPAIO, Cleuton; **Guia do Java: Enterprise Edition 5: desenvolvendo aplicações corporativas.** Brasport, 2007**.**

SILVA, Maurício Samy; **jQuery – A Biblioteca do Programador Javascript – 3ª Edição.** Editora Novatec, 2014

TITTEL, Ed; NOBLE, Jeff; **HTML, XHTML E CSS PARA LEIGOS.** Starlin Alta Editora, 7ª edição, 2014.

WAZLAWICK Raul Sidney. **Engenharia de Software: Conceitos e Práticas.** Elsevier Editora Ltda, 2013.

MEIRELES, Manuel; **Sistemas de informação: Quesitos de excelência dos sistemas de informação operativos e estratégicos.** Editora Arte e Ciência, 2004.