O uso de um sistema web no gerenciamento de clinicas veterinárias: estudo de caso na Carinhoso Pet Shop

Arthur Cardoso Cavalleiro de Macêdo, Luiz Jorge Ferreira de Sousa

Desenvolvimento de Sistemas Web 2015 – Estácio – Campus de Belém – Belém – PA– Brasil

arthur_cavalleirodemacedo@hotmail.com, ljfspa@gmail.com

Abstract. *This article is a case study that aims to identify the benefits that web systems can offer in management and decision making in a veterinary clinic. Its focus is the use of the System at the Carinhoso Pet Shop clinic, located in the city of Belém (PA), which acts to care for small animals and wild animals. The data collection or analysis of requirements was done through interviews with clinic owners, veterinarians, clients and employees. The results show that the use of this type of system facilitates the access to information and operations of the same any time and anywhere of online form, having just to have an access to the internet. And the sharing of information on animals, diagnosis and treatments applied, to all who use the system.*

Resumo. *Este artigo é um estudo de caso que tem como objetivo identificar os benefícios que os sistemas webs podem oferecer no gerenciamento e tomadas de decisão em uma clínica veterinária. Tem como foco o uso do Sistema na clínica Carinhoso Pet Shop, localizada na cidade de Belém (PA), que atua no atendimento de pequenos e animais silvestres. A coleta de dados ou análise de requisitos, foi feita através de entrevistas realizadas com os donos da clínica, veterinários, clientes e funcionários da mesma. Os resultados encontrados mostram que o uso desse tipo de sistema facilita o acesso a informações e operações do mesmo a qualquer momento e em qualquer lugar de forma online, bastando para isso ter apenas um acesso à internet. E o compartilhamento de informações sobre os animais, diagnóstico e tratamentos aplicados, a todos que utilizam o sistema.*

1. Introdução

Empreendimentos como uma clínica veterinária, possuem dificuldades quanto ao gerenciamento dos seus serviços. Por possuir vários clientes, a agilidade e qualidade do atendimento são prejudicadas, uma vez que se geram muitos documentos contendo as informações sobre os pacientes (histórico de vida do animal, histórico de doenças, registros de vacinas, registro de atendimentos como: consultas e tratamentos aplicados, etc). Além disso, os clientes dificilmente possuem acesso às informações que são registradas durante a visita ao veterinário, como por exemplo, o diagnóstico de seus animais e tratamentos aplicados, ou perdem documentos importantes, como a carteira de vacinação. Situações estas que poderiam ser evitadas/corrigidas se as informações estivessem armazenadas em um local comum a ambos, de fácil acesso.

A computação em nuvem é uma tecnologia que vem se tornando uma tendência no mercado. Os serviços funcionam a partir de uma conexão com a internet, e é possível acessar aplicativos, acessar ao banco de dados e executar diversas operações em qualquer lugar e a qualquer hora, bastando ter um computador com acesso à internet e sem a necessidade de instalação de programas para realizar estas tarefas. Esta tecnologia pode ser entendida como um ambiente de computação formado por diversos servidores sejam esses virtuais ou físicos, ou um conjunto de serviços com capacidade de processamento, armazenamento, aplicações, plataformas e serviços disponibilizados na internet (TAURION, 2009).

Mediante a isto, a justificativa do artigo é desenvolver uma solução em nuvem para otimizar e agilizar o acesso as informações de forma online, consequentemente a clínica poderá gerenciar o controle de serviços, vacinas, agenda do paciente (animal), agenda do veterinário e a vida de um animal em uma clínica veterinária e dar mais dinâmica ao processo de atendimento da clínica junto aos seus clientes.

2. Objetivos

2.1. Geral

Aperfeiçoar o gerenciamento de uma clínica veterinária, visando melhoria e facilidade no atendimento ao cliente, através da criação de uma solução que tem como base o que foi levantado acerca dos requisitos e necessidades apresentadas pela empresa. A mesma deverá ser simples, rápida e de fácil manuseio. Tendo como objetivo aperfeiçoar o gerenciamento de uma clínica veterinária, visando melhoria e facilidade no atendimento aos clientes, usuários, e veterinários.

2.2. Específicos

Gerenciar o controle de serviços, controle de vacinas e a vida de um animal em uma clínica veterinária; através do desenvolvimento de um sistema web, otimizando assim o processo de atendimento da clínica. Este sistema deverá ser ágil e que atenda todas as necessidades tanto da clínica quanto do cliente. De acordo com o levantamento de mercado, não foi visto tantas clínicas com um sistema web para atender seus usuários (veterinários, clientes) de forma online, onde os mesmos não necessitam ir até a clínica para executar determinadas operações.

Disponibilizar para os clientes, veterinários e para os usuários relatórios informações importantes necessárias para realizar as consulta e serviços de forma rápida e mais concreta, ter um controle de atendimentos, consultas, vacinas e tratamentos aplicados. Sendo assim, cada relatório mostrará todo o histórico de vida do paciente (animal), histórico de doenças, identificação das principais doenças que levaram o cliente a procurar ajuda na clínica, levantamento de vacinas, quantos pacientes já foram diagnosticados com determinada doença, gerar gráficos com informativos importantes como diagnósticos de doenças, relatório mensal de quantos pacientes foram atendidos.

Com isso o atendimento ao cliente será muito mais ágil e satisfatório tanto para o cliente quanto para à clínica que poderá ter um maior controle de suas consultas e o veterinário que poderá consultar sua lista de pacientes podendo também fazer a

remarcação de consultas e serviços caso não seja possível o mesmo comparecer a clínica.

3. Metodologia

A análise de requisitos foi feita através de entrevistas com donos da clínica, os veterinários e clientes, para identificar os problemas encontrados na forma de trabalho dos mesmos e assim conhecer as necessidades de todos para decidir como o sistema seria desenvolvido. Após esta análise se definiu que o sistema seria desenvolvido utilizando como arquitetura o padrão MVC, para garantir a independência entre as camadas, onde o sistema seria gerado através do *jhipster*, tendo assim um tempo bem menor no desenvolvimento e maior segurança na aplicação gerada, e a linguagem utilizada seria o *javascript*, que é a linguagem utilizada pelo *jhipster*, e o mesmo seria hospedado na plataforma de nuvem *heroku*, para facilitar o acesso ao sistema a qualquer momento.

4. Fundamentação Teórica

4.1 MVC

O modelo MVC (*model view control*) é um padrão de arquitetura de software que visa separar suas aplicações em 3 camadas: a camada de interação (*view*), a camada de manipulação de dados (*model*) e a camada de controle (*control*) (MACORATTI, s.d); a ideia de criar um modelo MVC em um projeto é permitir a todo e qualquer usuário acessar e visualizar todas as interfaces do programa em qualquer lugar e por várias pessoas, sem interferir nas demais atividades dos outros usuários que estão acessando o mesmo projeto (ARAÚJO, 2016).

4.2 Diagramas de casos de uso

O diagrama de casos de uso é utilizado para melhor representar o tipo de comunicação que ocorrerá entre os desenvolvedores do sistema e o cliente, onde o mesmo mostrará quais as funcionalidades que o seu sistema terá ao ponto de vista do usuário e dar a sua opinião ou acrescentar algo mais em sua estrutura. O diagrama de casos de uso é representado por: atores, que pode ser tanto um usuário do sistema, quanto um usuário computacional; Casos de uso, que definem qual a função do sistema e de cada ator que estará relacionado à função ao ponto de vista do usuário; Relacionamentos, que ajudam a descrever qual o tipo de interação entre o usuário e a estrutura, seja entre o ator e o casos de uso, ator “a” com Ator “b” ou entre casos de uso, que é representada pelo *include* e o *extend* (GUEDES, 2011).

4.3 Diagrama de Classe

O diagrama de classe é um diagrama utilizado para descrever os principais objetos e os relacionamentos entre eles. O diagrama de classe representa a estrutura estática do projeto e como ela está sendo modelada, focando nos elementos do sistema,

que contém as entidades, que visa representar a classe concreta e a classe abstrata, ambas com características iguais, contendo o nome da classe, seus atributos e seus métodos podendo o cliente visualizar os parâmetros de cada método, inclusive os tipos de cada um, os tipos de atributos e os valores de retorno de cada método; e seus Relacionamentos, onde é mostrado quem está se relacionando com quem, através de uma representação de navegabilidade para que o usuário compreenda melhor quem será o responsável por cada atividade (BEZERRA, 2007).

4.4 Modelo Entidade Relacionamento

O modelo entidade relacionamento é um modelo conceitual utilizado para descrever os objetos (entidades) envolvidos no modelo, com suas características (atributos) e como elas se relacionam entre si (relacionamento). Este modelo nada mais é do que a representação de como será o banco de dados da aplicação, onde incidem sobre as relações dos elementos dentro de uma entidade (ALEXANDRUK, 2011).

4.5 JAVASCRIPT

O *javascript* é utilizado para se ter uma melhor visualização das funcionalidades de um sistema, que nada mais é do que uma série de instruções escritas para serem seguidas. É uma linguagem pequena e leve que pode ser facilmente incorporada em qualquer aplicação web. O *javascript* é uma linguagem muito usada hoje para sistemas do tipo *home-page*, e é orientada a objetos e foi criada pela *netscape* visando expandir a funcionalidade dos sistemas web. É uma linguagem que independe de plataformas, seja *windows, linux, macintosh*, etc. Depende apenas do navegador que irá interpretar o código, que será executado sem que nenhuma adaptação seja necessária nesse navegador. *Javascript* no lado do cliente mostra a linguagem básica através do fornecimento de objetos para o seu navegador e sua documentação, ou seja, permite que o cliente faça uma aplicação de elementos dentro de um formulário *html* e responderá o usuário através de cliques de mouse, entrada de dados e navegação da página (SOUZA, 2016).

4.6 Jhipster

Jhipster é um *framework* baseado em linha de comando para a criação de toda a estrutura do sistema que se denomina de *scaffolding*, que nada mais é do que o esqueleto do projeto. Todo o *framework* necessita de seu *scaffolding*, e não é diferente no *jhipster*, onde o seu "esqueleto" é o *yeoman*, ele tem foco no *front-end*, pois assim ele gera toda a estrutura do projeto de forma otimizada e ágil, e faz o seu trabalho tanto na parte do *back-end* como no *front-end*. Além disso, ele gera todas as classes *java* e arquivos de configurações para a necessidade do projeto. Vale destacar também que a customização é muito relevante, pois é possível criar vários projetos com diferentes tipos de setups, como *maven* ou *gradle*, banco relacional ou não relacional, e conta também com diversas opções de tecnologias, tanto para o cliente quanto para o servidor (RAIBLE, 2016).

4.7 Heroku

Heroku é uma plataforma em nuvem como um serviço (PaaS) que suporta várias linguagens de programação que são usadas como um modelo de implantação de aplicativos da web. A *heroku*, foi uma das primeiras plataformas em nuvem, está em desenvolvimento desde junho de 2007, quando era só oferecido suporte à linguagem de programação *ruby*, mas agora é compatível com *java*, *node.js*, *scala*, *clojure*, *python*, *php* e *Go*. E por isto o *heroku* é conhecido como uma plataform poliglota, pois permite ao desenvolvedor criar, executar e escalar aplicativos de forma semelhante em todas as linguagens (PINHO, 2016).

5. Estudo de Caso

5.1. Cenário atual

Atualmente as clínicas contam com o auxílio de planilhas, anotações, agendas, e isso dificulta e atrasa um pouco quando o cliente liga para solicitar algum serviço, pois ainda tem que pesquisar qual o tipo de serviço, se tem veterinário livre para atendê-lo, se não há outra marcada para a mesma hora que o cliente está solicitando. Tudo isso implica em uma série de problemas tanto para a empresa quanto para o cliente, que quer mais rapidez e eficácia no atendimento e nas respostas de serviço;

5.2. Cenário Proposto

A solução proposta foi desenvolver um sistema que atendesse a toda e qualquer necessidade tanto da clínica, quanto do cliente a qualquer momento. Com base no que foi levantado, foi feito algo que acelere o atendimento da clínica, caso o agendamento de serviço (como banho, tosa, consultas) seja feito, neste momento o sistema deve gerar um relatório de serviço; onde o usuário já verá quais são os serviços que estão agendados para a semana e a partir disso marcar consultas para o cliente. Onde o cliente poderá decidir como fazer tudo: via web, telefone móvel, e o mesmo terá a opção de fazer os agendamentos de serviços online informando qual o tipo de serviço e solicitando também a forma de pagamento que será efetuado.

O sistema web terá o intuito de cadastrar o veterinário, cadastrar o dono do animal (que nesse caso será o nosso cliente), cadastrar o animal, cadastrar consultas, cadastrar agendamentos, cadastrar perfis de acesso aos usuários e aos clientes, cadastrar os tipos de serviços, gerar relatório semanal ou mensal de serviços, gerar relatório do histórico de vida do animal, gerar relatório de histórico de consultas, gerar relatório de atendimentos, onde o mesmo será acessado tanto pelo usuário, quanto pelo cliente a qualquer momento, sem precisar comparecer na empresa para a realização de tal.

Usamos o diagrama de caso de uso mostrado na figura 01; que possui os seguintes Atores.

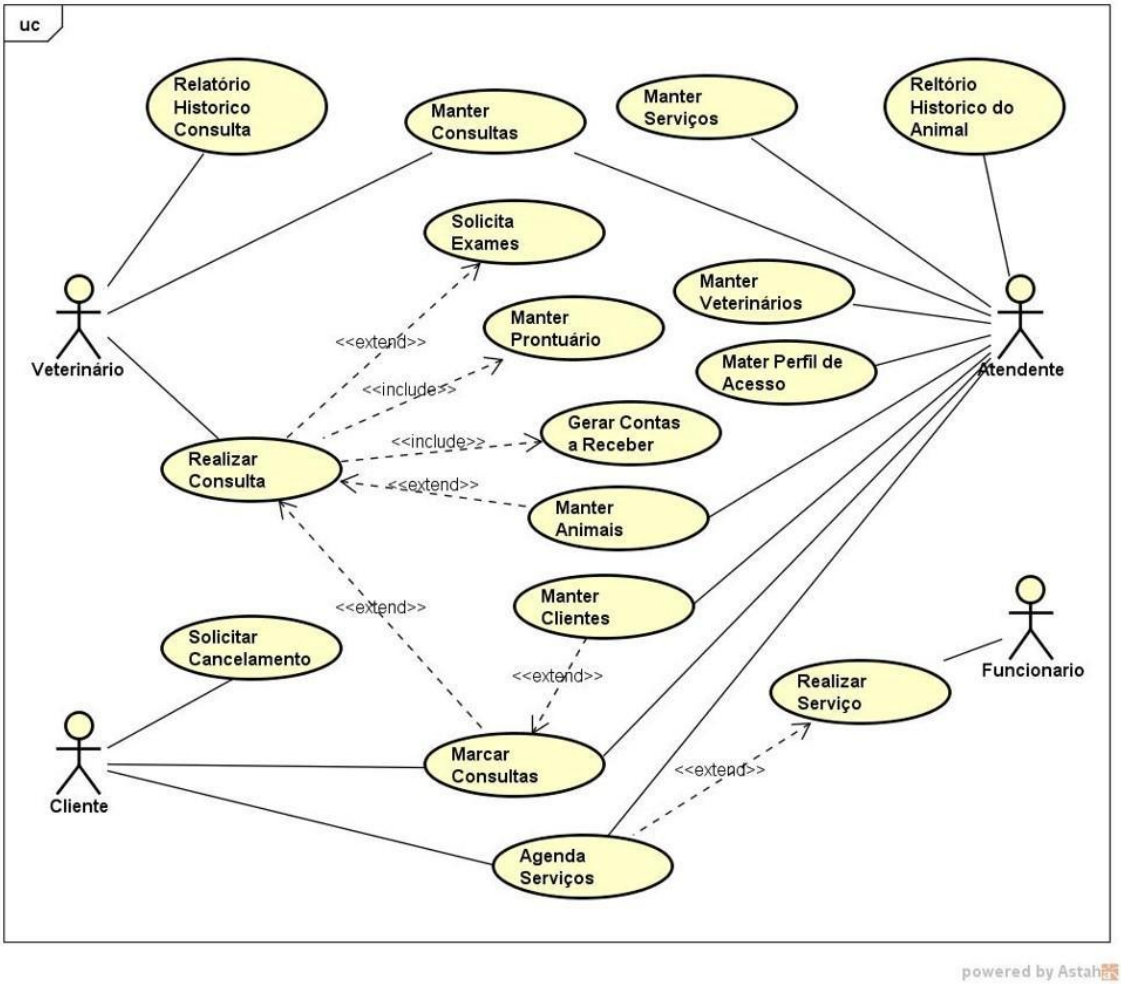
- Veterinário, Cliente, Funcionário e Atendente onde:
- Veterinário: Médico responsável pelas Consultas e Tratamentos Aplicados;

- Cliente: Pessoa que solicita Serviços; como Tosa e Banho; para seus Animais;
- Funcionário: Pessoa responsável por várias operações do sistema na clínica como Administração do sistema e Gerência da clínica;

Onde esses atores podem executar as seguintes ações:

- Manter consultas;
- Solicitar exames;
- Emitir relatórios e consultas;
- Fazer agendamentos de serviços e consultas;
- Executar serviços como tosa e banho;
- Reagendar consultas e serviços;
- Gerenciar usuários;
- Controle de recebimentos;
- Controle de estoque;
- Controle de acesso ao sistema;
- Controle de serviços como banho e tosa.

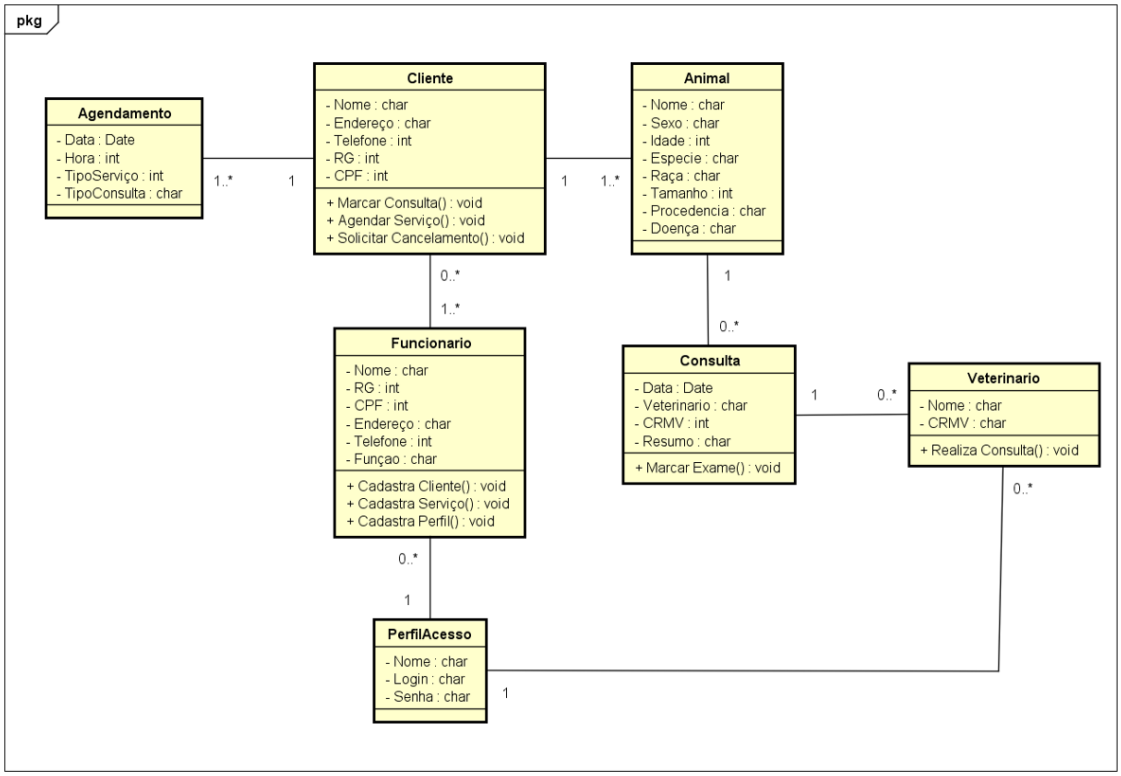
Figura 01 - Diagrama de Caso de Uso



O diagrama de classe que mostra os principais objetos e os relacionamentos entre os mesmos no sistema. E é mostrado na Figura 02, onde os objetos e relacionamentos são:

- 1 para N
  - Cliente / Agendamento
  - Cliente / Animal
  - Veterinário / Consulta
  - Funcionario / Agendamento
- 1 para 1
  - Animal / Cliente
  - Funcionario / PerfilAcesso
  - Cliente / PerfilAcesso
  - Animal / Raca
  - Animal / Especie
- N para N
  - Funcionario / Funcionario

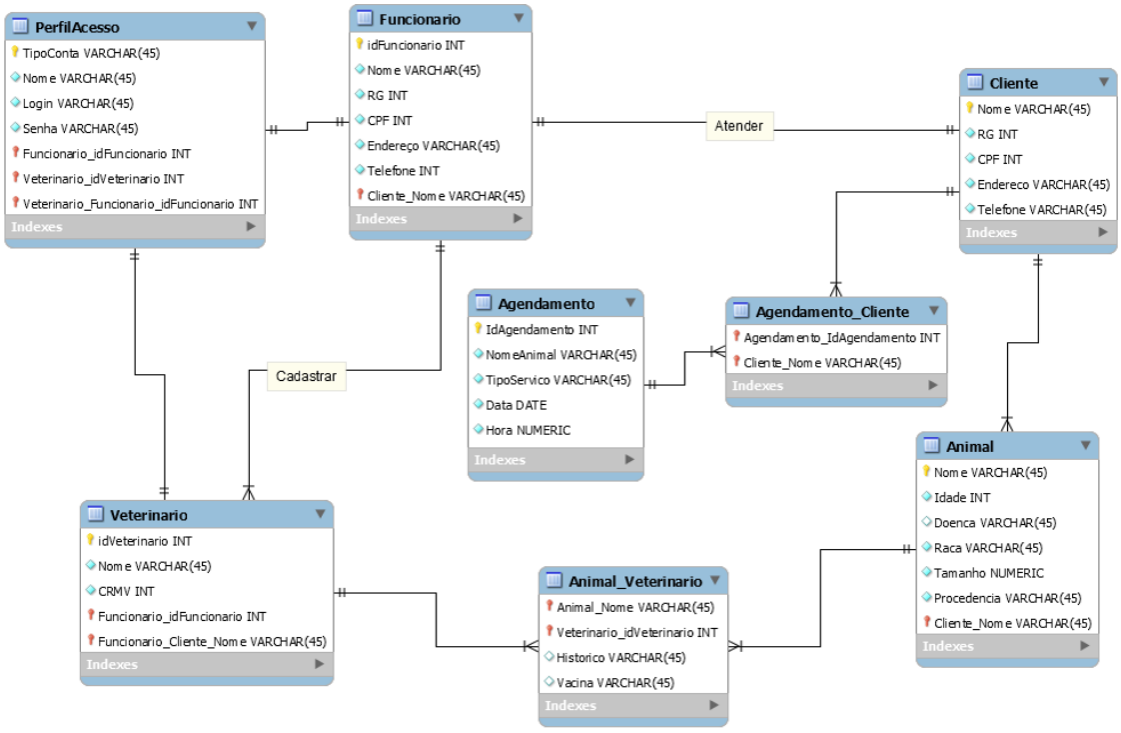
Figura 02 - Diagrama de Classe



O modelo entidade relacionamento utilizado, mostra as seguintes entidades com seus atributos e é o mostrado na figura 03, com as seguintes entidades:

- Perfil de Acesso
- Funcionário
- Veterinário
- Cliente
- Agendamento
- Agendamento Cliente
- Animal
- Animal Cliente
- Animal Veterinário
- Serviço
- Tratamento

Figura 03 - Modelo Entidade Relacionamento



5. Resultado

Como resultado foi criado o sistema *ClinicVet* que está hospedado em nuvem através do *heroku* e para fazer o acesso através do endereço *https://clinicvet.herokuapp.com/*, facilitando assim o acesso às informações que estão contidas no sistema, onde o acesso ao mesmo é feito de forma mais rápida e otimizada e diminuindo o tempo de resposta tanto para o cliente, que caso ele necessite realizar agendamento de algum serviço, ou consultar os relatórios com todas as informações que se deseja saber de seu animal, para o médico veterinário, que deseja consultar quantas consultas ele realizará no dia, ou na semana, grau de importância e necessidade de cada paciente, solicitar a ajuda de algum outro médico, caso necessário, reagendar algum paciente quando o médico ou o mesmo não puderem comparecer, realocar um médico para suprir a falta dele, quanto para o próprio usuário, que consulta toda a sua agenda, marcações de agendamentos, espaço livre para novas consultas, verificar disponibilidade de algum médico, caso o cliente tenha algum médico que prefira ser atendido.

Com o sistema é possível obter vários relatórios, como:

- Cadastro:
    - Funcionários;
    - Veterinários;
    - Clientes;
    - Animais;
    - Atendimentos;
    - Diagnósticos / Tratamentos;
    - Procedimentos;
    - Agenda de Vacinas;
    - Vermífugos;

    - Serviços;
    - Produtos.
- Acompanhamento clínico;
- Agenda de resultado de exames que estão nos laboratórios;
- Todos os retornos e compromissos agendados;
- A Ficha de Cadastro do animal e do seu proprietário;
- Vacinas - inclusive com data de revacinação;
- Consulta - com diagnóstico e terapêutica aplicada;
- Informações sobre a terapêutica administrada - medicamentos, quantidades e horários que estão sendo administrados para o animal;
- Gestão de serviços como Banho e Tosa;
- Histórico clínico do Animal;
- Validade de Produtos como Remédios e Vacinas em estoque;
- Evolução Clínica de cada Paciente;
- Carteira de Vacinação;
- Validação de Produto como vacinas, medicamentos e ração;
- Relatório de caso clínico.

Nas figuras 4 e 5 são mostrados os exemplos alguns relatórios.

Figura 04 – Exemplo 1

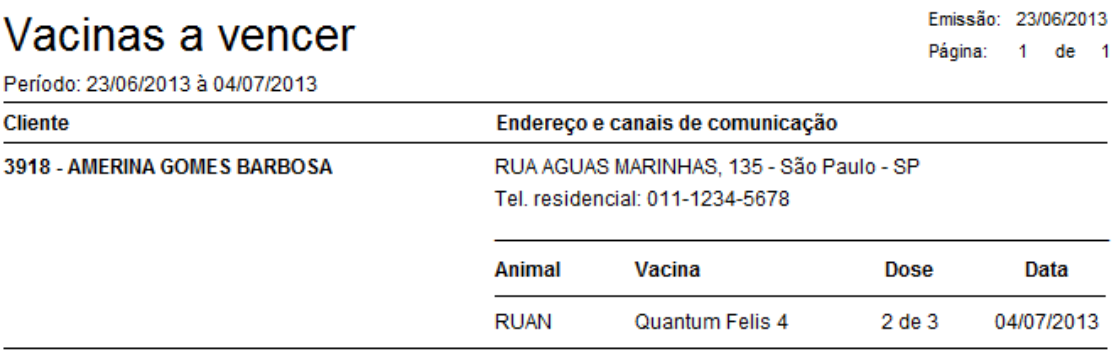
Lista de agendamentos para transporte

Emissão: 27/04/2013
Página: 1 de 1

Lista completa classificada por hora de início

Hr ini. / Hr fin.	Código do Cliente / Código do Animal	Apelido/Nome Fantasia / Nome do animal	Raça do animal	Nome/Razão Social / Espécie do animal	Endereço / Tipo de serviço / Observação do agendamento	Nome do serviço
10:00	1186	ADALBERTO		ADALBERTO MARCELINO DA SILVA JUNIOR	RUA CORONEL XAVIER CHAVES, 275 A - Centro - São Paulo	
10:30	1310	LUKE	S.R.D	Canino	Banho, tosa e Estética	BANHO
10:30	790	ALBERTO		ALBERTO DISPATO	RUA JOSE CABRAL SILVEIRA, 198 - Centro - São Paulo - SP	
11:30	858	MENINA	S.R.D	Canino	Banho, tosa e Estética	BANHO PORTE GRANDE

Figura 05 – Exemplo 2



Vacinas a vencer

Emissão: 23/06/2013
Página: 1 de 1

Período: 23/06/2013 à 04/07/2013

Cliente	Endereço e canais de comunicação
3918 - AMERINA GOMES BARBOSA	RUA AGUAS MARINHAS, 135 - São Paulo - SP Tel. residencial: 011-1234-5678

Animal	Vacina	Dose	Data
RUAN	Quantum Felis 4	2 de 3	04/07/2013

6. Conclusão

A criação do sistema com o *jhipster* agilizou o processo de criação do mesmo, pois o mesmo levou um tempo de criação de aproximadamente de dois dias e na forma tradicional levaríamos em média uns 3 meses.

Outro ponto encontrado em relação a um sistema desenvolvido na plataforma web foi a facilidade de acesso ao mesmo por todos, pois para isto basta ter um acesso a internet e um navegador; onde se informe o endereço do mesmo.

Como este sistema está hospedado em nuvem, o mesmo no futuro se tornará uma vasta biblioteca de conhecimento sobre administração de clínicas e também sobre doenças e aplicação de tratamentos para todos os que fizerem uso do mesmo, compartilhando assim a informação com todos.

7. Referências Bibliográficas

ARAÚJO, E. C. **ASP.NET MVC 5** – Crie aplicações web na plataforma Microsoft. Casa do Código. 2016.

ALEXANDRUK, M. **Modelagem de Banco de Dados**. 2011, Disponível em: < www.unilivros.com.br/pdf/dbmod.pdf>. Acesso em: Agosto 2017.

BEZERRA, E. **Princípios de análise e projeto de sistemas com UML**. Editora Campus. 2007

PRATES, L. **Introdução ao Framework JHipster**. Disponível em: < http://www.devmedia.com.br/introducao-ao-framework-jhipster/34043>. Acesso em: Abril 2017.

GUEDES, G. T. A. **UML 2** : uma abordagem prática. 2. ed. São Paulo : Novatec Editora, 2011.

MACORATTI, J. C. **Padrões de Projeto:** O modelo MVC - Model View Controller. Disponível em: <www.macoratti.net/vbn_mvc.htm>. Acesso em: agosto 2017.

MELO, A. C. **Desenvolvimento aplicações com UML 2.0:** do conceitual à implementação – 2. Ed. – Rio de Janeiro : Brasport, 2004.

PINHO, D. M. **Deployando seu projeto em node.js no Heroku**. 2016, Disponivel em: < https://medium.com/code-prestige/deployando-seu-projeto-em-node-js-no-heroku-b49a6ae7dbc3 >. Acesso em: Agosto 2017

RAIBLE, M. **The JHipster Mini-Book 2.0**. InfoQ. 2016

SOUZA, A. **Java EE** – Aproveite toda a plataforma para construir aplicações. Casa do Código. 2016.

TAURION, C. **Cloud Computing: computação em nuvem: transformando o mundo da tecnologia da informação**, Editora Brasport: Rio de Janeiro, Brasil, 2009.