CENTRO UNIVERSITÁRIO UNIFACVEST
CURSO DE CIÊNCIA DA COMPUTAÇÃO
TRABALHO DE CONCLUSÃO DE CURSO
FELIPE DONIZETE MACEDO

GERENCIAMENTO DE HOSPITAIS VETERINÁRIOS

UNIVET

LAGES

2013

FELIPE DONIZETE MACEDO

GERENCIAMENTO DE HOSPITAIS VETERINÁRIOS

UNIVET

Projeto apresentado à Banca Examinadora do Trabalho de Conclusão do Curso de Ciência da Computação para análise e aprovação.

LAGES

2013

FELIPE DONIZETE MACEDO

GERENCIAMENTO DE HOSPITAIS VETERINÁRIOS

UNIVET

Trabalho de Conclusão de Curso de Ciência da Computação apresentado ao Centro Universitário UNIFACVEST como parte dos requisitos para obtenção do título de bacharel em Ciência da Computação.

Prof. MSc. Márcio José Sembay.

Lages, SC ____/____/2013. Nota_____ _____

_____

_____

_____

LAGES

2013

AGRADECIMENTOS

Agradeço primeiramente a Deus por me dar forças para continuar e seguir em frente com os meus objetivos, e me iluminar durante toda essa trajetória;

Aos meus pais pelo incentivo, apoio, no qual foi fundamental para que eu pudesse dar continuidade aos meus estudos, que sempre me auxiliaram em todos os momentos;

Aos professores por transmitir seus conhecimentos durante todo o curso, e ao meu orientador Marcio Sembay pelo acompanhamento e por acreditar e acompanhar meu trabalho.

SUMÁRIO

LISTA DE ABREVIATURAS

TI – Tecnologia da Informação

HTML – HyperText Markup Language

JVM – Java Virtual Machine

POO – Programação Orientada a Objetos

SGDB – Sistema Gerenciador de Banco de Dados

B.I - Business Inteligence

MVC – Model View Controller

W3C - World Wide Web Consortium

HTTP - Hyper Text Transport Protocol

SGBD - Gerenciador de Banco de Dados

SQL - Structured Query Language

TB - Twitter Bootstrap

SHI - Sistema de Informação Hospitalar

UML - Unified Modeling Language

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Este trabalho apresenta um sistema na linguagem de programação PHP, com o objetivo de gerenciar o hospital veterinário UniVet. Para atingir essa meta foram realizadas pesquisas bibliográficas na área de sistemas veterinários e programação web. Para atender as necessidades requisitadas, foi idealizado o UniVet Gerenciamento de Hospital Veterinário. Com o UniVet, será possível armazenar os dados dos animais e seus donos através de cadastros em banco de dados e acessá-los de forma online. Com a análise de requisitos foi possível abstrair as necessidades e problemas encontrados. Com base no levantamento de dados, foram feitas pesquisas bibliográficas referentes a padrões de desenvolvimento e aplicações de gerenciamento de atividades, buscando-se desta forma atingirem o resultado esperado na utilização do sistema de gerenciamento do hospital veterinário da universidade.

Palavras chave: Sistema de informação. Hospital veterinário. Websites. UniVet. Gerenciamento.

ABSTRACT

This paper presents a system in PHP programming language, in order to manage the veterinary hospital Univet. To achieve this goal literature searches were conducted in the area of veterinary systems and web programming. To meet the needs requested, was designed to Univet Management Veterinary Hospital. With Univet, you can store the data of the animals and their owners through records in the database and access them online so. With the requirements analysis was possible to disregard the needs and problems encountered. Based on survey data, literature searches were made relating to standards development and application management activities, thereby seeking to achieve the expected result in the use of management system of the university veterinary hospital.

Keywords: information system. Veterinary Hospital. Websites. Univet. Management.

I. INTRODUÇÃO

Os programas de computadores foram projetados para solucionar problemas humanos, com esta visão existe um processo chamado de programação dinâmica, a qual consiste em uma técnica para divisão de problemas maiores em partes menores (SANDERS, 2013).

Segundo Chicoli (2008), para a construção de conteúdo para internet, é importante estar atualizado a respeito de tendências, recursos novos, ferramentas e tudo mais o que a rede oferece todos os dias. No inicio da Internet os sites apenas consistiam em simples coleções de texto, hoje a utilização de alguns recursos é determinante para que um site possa ser mais ou menos atraente, as pessoas necessitam interagir com o conteúdo, ter influencia com o que está publicado, esta fase é chamada de Web 2.0, onde seu fundamento consiste na interação das pessoas com a ferramenta web, seja com o contato através de formulários, atualização dinâmica de conteúdos, até a tomada de decisão através de sistemas de B.I (*Business Inteligence*).

> A web 2.0 não apresenta nenhuma mudança tecnológica significativa, mas uma mudança de foco. Começou uma percepção de que os Websites deveriam se integrar, deixando de ser estanques e passando a trocar conteúdo (SAMPAIO, 2007, p. 8).

Desenvolver um sistema web possibilita uma maior mobilidade de acesso ao usuário, pois o mesmo pode ser acessado por qualquer dispositivo que possua tecnologia para navegação na internet, seja este um desktop, dispositivos móveis ou qualquer outro que seja compatível com o HTML.

Na tentativa de produzir softwares com qualidade e de fácil manutenção, teve-se a necessidade de ser desenvolvido o Paradigma Orientado a Objetos para desenvolvimento de um software, este paradigma esta embarcado na Engenharia de Software, sendo que pode ser apresentados alguns itens como metas de engenharia de Software relacionada a orientação a Objetos, como o Aumento do encapsulamento, aumento da ocultação de informações com o uso de um objeto, aumento da reusabilidade do código, diminuição de custos e do tempo de desenvolvimento, facilita na manutenção, tanto corretiva quanto evolutiva (ENGHOLM JUNIOR, 2010).

Abordar o desenvolvimento de software como engenharia e aliar o assunto com ferramentas e técnicas voltadas ao aumento da produtividade e qualidade dos artefatos gerados foi uma solução encontrada. Devido ao alto custo do desenvolvimento de software e à sua importância cada vez maior, a engenharia de software passa a ter um valor muito grande na garantia de sistemas eficientes e de menor custo (ENGHOLM JUNIOR, 2010).

Para uma aplicação web deve-se tratar da complexidade valendo-se de componentes modulares, no qual se deve encapsular tudo de que o componente precisa, dentro de partes pequenas e bem definidas da aplicação, fazendo que permita a divisão de uma grande aplicação. Na criação de aplicações de forma modular desde o princípio, estará oferecendo uma fundação sólida capaz de suportar um possível crescimento futuro (LOUDON, 2010).

Para a construção de um sistema web é necessário analisar vários fatores, é imprescindível, que antes de iniciar a implementação deste, sejam realizadas etapas como: Levantamento de requisitos, estudo de viabilidade, análise de caso de uso e modelagem de dados. Já na fase de implementação é importante adotar técnicas e padrões de desenvolvimento, garantindo assim, maior flexibilidade para geração de códigos e eficiência na usabilidade do software assegurando a qualidade para o usuário final do sistema.

O modelo de arquitetura de software MVC (*model-view-controller*), vem sendo adotado como uma solução reutilizável para implementação de sistemas, pois com base na mesma estrutura, é possível implementar diversos sistemas. O emprego de um *framework* no desenvolvimento de um software segue o mesmo conceito de MVC, pois, promove a reutilização de códigos, servindo de base para vários projetos distintos (GABARDO, 2012).

> CodeIgniter é um *framework* de aplicações web de código aberto para a linguagem PHP. CodeIgniter tem muitas características que o destacam da multidão. Ao contrário de alguns outros frameworks PHP que você pode ter vindo através, a documentação é muito profunda e completa, cobrindo todos os aspectos da estrutura. CodeIgniter também será executado em ambientes de hospedagem compartilhada, pois tem uma pegada muito baixo, mas ainda tem um desempenho excepcional (GRIFFITHS, 2010, p. 17).

Atualmente o Hospital Veterinário do Centro Universitário Unifacvest tem a necessidade de ter um sistema de gerenciamento de seus processos. Para atender esta necessidade, idealizou-se o UniVet Gerenciamento de Hospital Veterinário. Com o UniVet, será possível armazenar os dados dos animais e seus donos através de cadastros em banco de

dados e acessá-los de forma online, fazendo o uso de rotinas como agendamento de consultas, roteiro de vacinas e acompanhamento médico com histórico de ocorrências, otimizando assim o gerenciamento das atividades do hospital veterinário da instituição.

Com a análise de requisitos foi possível abstrair as necessidades e problemas encontrados. Com base neste levantamento, foram feitas pesquisas bibliográficas referentes a padrões de desenvolvimento e qualidade de software, implantação de sistemas *Web* e aplicações de gerenciamento de atividades, buscando-se desta forma atingirem o resultado final esperado na utilização do sistema de gerenciamento de processos do hospital veterinário da universidade.

> Podemos definir requisito como uma condição ou capacidade de um software que deve ser implementada por um sistema ou componente de sistema para se alcançar determinado fim. Todo projeto de software tem um conjunto de requisitos, definidos pelas necessidades e expectativas dos usuários que efetivamente utilizarão o mesmo, relacionado ao atendimento dos objetivos de negócio da empresa onde trabalham (JUNIOR, 2010, p. 151).

1.1 Justificativa

Segundo Engholm Jr. (2010), um software tem uma importância cada vez maior no dia-a-dia das empresas, devido a isso, devemos nos preocupar com a maneira que ele agrega valor aos negócios das mesmas, aumentando assim a produtividade e diminuindo os custos.

> Todo projeto de software é iniciada por alguma necessidade do negócio - a necessidade de corrigir um defeito em uma aplicação existente, a necessidade de adaptar um sistema legado para mudar o ambiente do negócio, a necessidade de estender as funções e características de uma aplicação existente, ou a necessidade de criar um produto novo, serviço ou sistema (PRESSMAN, 2006, p. 12).

Com embasamento no estudo de caso realizado para a implantação do UniVet no hospital veterinário Facvest, foram encontradas algumas necessidades, que seriam solucionadas eficientemente com a utilização de um sistema de informação.

Todos os processos do hospital eram feitos de forma manual, as fichas dos pacientes eram documentadas em papel, e não tinham um local específico para armazenamento, desta forma, não era possível ter um controle sobre o histórico de atendimento e sobre as próximas vacinas dos animais, isso sem considerar que possuir os documentos em forma digital, traz benefícios como, agilidade, segurança das informações e maior disponibilidade para acesso através da web.

A implantação do UniVet será de grande utilidade para o hospital veterinário, a implantação do mesmo irá auxiliar no controle de atendimentos dos pacientes, com a disponibilidade de recursos como roteiro de vacinas, agendamento de consultas e até mesmo histórico dos pacientes. Com isto, a instituição terá maior controle sobre as atividades do hospital veterinário, pois uma vez que todas estas informações estarão armazenadas em banco de dados, será viável a geração de relatórios e abstração de dados dos pacientes.

1.2 Importância

Na implantação de um sistema, a última fase do desenvolvimento de um software é a parte do treinamento e documentação, essa fase é chamada de "prova de fogo", sendo ele aceito ou não pelo usuário final e pelas partes que serão responsáveis por manter este sistema, seja prestando suporte, dando treinamento ou apenas gerenciando. O principal objetivo desse projeto é a garantia que o sistema ou software seja implantado com sucesso e que a satisfação do cliente ou usuário seja plena (REZENDE, 2006).

Informatizar o hospital veterinário da Unifacvest trará benefícios que vão além das questões de segurança e disponibilidade da informação. A implantação do UniVet aumentará a eficiência nos diagnósticos prestados aos pacientes, possibilitando assim a identificação de doenças crônicas, melhor acompanhamento dos acadêmicos do curso de veterinária, maior segurança nos atendimentos prestados, uma vez que não necessariamente, o mesmo aluno ou instrutor será o responsável pelo atendimento de um mesmo paciente.

1.3 Objetivos do Trabalho

1.3.1 Objetivo Geral

Desenvolver um sistema Web para controle e gerenciamento de processos do Hospital Veterinário Facvest, possibilitando um melhor atendimento ao cliente e armazenamento de históricos dos pacientes.

1.3.2 Objetivos Específicos

Os Objetivos Específicos consistem no desenvolvimento de um sistema de gerenciamento de processos do hospital veterinário Facvest com as seguintes funcionalidades:

a) Cadastro de clientes.

b) Cadastro de pacientes.

c) Agendamento de consultas.

d) Impressão de Prontuários.

e) Lembretes.

f) Controle de nível hierárquico de usuários.

Através destas funcionalidades os usuários poderão gerenciar informações de animais e proprietários, agendamentos, vacinas, dentre outras.

1.4 Metodologia

1.4.1 Estudo de Caso

Para realizar o desenvolvimento deste trabalho foi realizada uma entrevista com a coordenadora do curso de Veterinária da UniFacvest para avaliação de requisitos, onde foram

levantadas as necessidades do sistema e a qual ponto este iria auxiliar no processo de gerenciamento do hospital veterinário.

1.4.2 Estudo Bibliográfico

Realizou-se um estudo sobre padrões de desenvolvimento em aplicações de grande porte, efetuou-se pesquisa bibliográfica em livros e periódicos sobre desenvolvimento web, onde foram apontadas algumas metodologias de desenvolvimento, e como estas auxiliariam no desenvolvimento e na codificação do UniVet, garantindo-se assim a ergonomia e usabilidade do sistema.

1.4.3 Cronograma

O seguinte cronograma foi utilizado para o desenvolvimento deste trabalho.

Atividades Realizadas	**Agosto**	**Setembro**	**Outubro**	**Novembro**
Pesquisa				
Pré-Proposta				
Modelagem do sistema				
Implementação do software				
Entrega e defesa do TCC à banca avaliadora				
Correção				

Tabela 1 - Cronograma do TCC 1
Fonte: O próprio autor

1.4.4 Estrutura do Trabalho

Para o desenvolvimento deste trabalho de conclusão de curso, foram necessárias as seguintes etapas: pesquisa de material bibliográfico, revisão bibliográfica, modelagem do sistema, implementação do software e testes.

Na primeira etapa foi realizado um levantamento bibliográfico, coletando informações e dados para o início da realização do trabalho. Na revisão bibliográfica fundamentou-se teoricamente justificando os tópicos abordados neste trabalho. A modelagem de sistema foi realizada com a ferramenta Netbeans e com os padrões requeridos pela 3WC. Foram feitas pesquisas em livros, periódicos, páginas da Web, entre outras modalidades.

II. FUNDAMENTAÇÃO TEÓRICA

2.1 Web

A Web foi inventada no ano de 1992 por Sim Tim Berners-Lee, que atualmente é diretor do *World Wide Web Consortium* (W3C). O W3C é uma organização que desenvolve padrões e normas de desenvolvimento (SILVA, 2011).

A Web pode ser definida como um sistema de documentos que são interligados e executados na internet, onde os dados que são disponibilizados na Web podem ser acessados através de browsers também conhecidos como navegadores de internet. Estes documentos podem estar em forma de vídeos, fotos, sons, hipertextos entre outros.

> A Web é um espaço de informação abstrato (imaginário). Na Internet você encontra computadores – na Web, você encontra documentos, sons, vídeos, informação. Na internet, as conexões são cabos entre computadores; na Web, as conexões são os links de hipertextos. A Web existe devido a programas que se comunicam entre computadores na Internet. A Web não poderia ser criada sem a Internet. A Web se tornou a rede útil porque as pessoas estão realmente interessadas em informação (para não citar conhecimento e sabedoria!) e realmente não querem saber sobre computadores e cabos (ARAYA; VIDOTTI, 2010, pag. 26).

Segundo Martín e Martín (2011), os conceitos de Internet e Web podem parecer o mesmo, mas são bem diferentes, sendo que a internet engloba as tecnologias que permitem que computadores em diferentes lugares do mundo possam compartilhar informações, em que essa conexão pode ser feita através de cabos, modems, linhas telefônicas e que a internet é uma forma de organizar a informação usando um meio físico de comunicação da rede de internet através de um protocolo HTTP. Este protocolo HTTP é um protocolo de transferência de hipertexto que os navegadores utilizam para realizar requisições aos servidores Web e receber requisições, este mesmo protocolo é utilizada em paginas Web.

Segundo Comer (2007), quando um navegador interage com um servidor Web os dois programas seguem o Hyper Text Transport Protocol conhecido como HTTP, que a principio permite que o navegador solicite um item específico, que o servidor então retorna os dados solicitados. O HTTP suporta quatro operações básicas que um navegador pode especificar quando faz um pedido, sendo eles o GET, HEAD, POST e o PUT.

2.2 Linguagens de programação web

Para o desenvolvimento web é necessário conhecer algumas linguagens de programação, e saber o beneficio que cada uma delas pode oferecer. Neste capitulo será abordado à linguagem de programação PHP e algumas linguagens que ela possui interação no desenvolvimento de conteúdos dinâmicos para a Internet, como o HTML, CSS, Javascript, entre outros.

2.2.1 PHP

A linguagem de programação PHP (*Hypertext Preprocessor*) foi concebida durante o outono de 1994 por Rasmus Lerdorf, sendo que suas primeiras versões não foram disponibilizadas, tendo sido utilizadas em sua homepage apenas para que ele pudesse ter informações sobre as visitas que estavam sendo feitas, a primeira versão a ser disponibilizada para outras pessoas foi no ano de 1995, onde o interpretador foi reescrito, ganhando o nome de PHP/FI, o FI como foi chamado era um pacote escrito por Rasmus que interpretava dados de formulário HTML (Form Interpreter), onde ele combinou os dados do pacote Personal Home Page Tools com o FI e adicionou suporte ao MySql. Estima-se que em 1996 o PHP/FI estava sendo usado por cerca de 15.000 sites pelo mundo, e em 1997 esse número subiu para mais de 50.000, nesta mesma época o PHP deixou de ser um projeto de Rasmus e passou a ter contribuições de outras pessoas para ter uma equipe de desenvolvimento mais organizada (MUTO, 2006).

O PHP é uma linguagem muito utilizada no desenvolvimento web, que tem um crescimento considerável no mercado comercial, em que podem ser citados alguns ambientes como o Facebook, Flickr, partes do Yahoo e Wikipedia, onde todos utilizam o PHP de modo significativo, e alguns sistemas de gerenciadores de conteudo, como o Drupal, Joomla e WordPress (MACINTYRE, 2010).

> [...] A principal diferença em relação às outras linguagens é a capacidade que o PHP tem de interagir com o mundo web, transformando totalmente os websites que possuem páginas estáticas (NIEDERAUER, 2011, p. 24).

Com o script PHP é possível concatenar com as tags HTML, que seria a linguagem de marcação, dessa forma é possível escrever vários trechos de código PHP em uma única página. Essa combinação entre HTML e PHP é muito útil, pois se utiliza o PHP para gerar os dados de forma dinâmica, enquanto o HTML é utilizado para formatar e exibir esses dados nas páginas mostradas no navegador (NIEDERAUER, 2008).

> O PHP é uma linguagem de programação interpretada pelo servidor que fica embutida junto ao código HTML. Quando o arquivo é disponibilizado através da Internet por um servidor de páginas capaz de gerenciar a execução de scripts PHP e algum cliente (browser) pede tal arquivo lhe seja enviado, o servidor de páginas checa se existe, dentro deste arquivo, algum código escrito na linguagem PHP (BRUNO et al., 2010, p.29).

Além de poder concatenar o PHP com páginas HTML também podem ser descritas algumas vantagens de utilizar esta linguagem de programação, entre algumas delas:

- O PHP é uma linguagem gratuita e *open source*, sua versão mais atual pode ser baixada pelo site oficial;
- Multiplataforma, ele pode ser usado em diversos sistemas operacionais;
- É uma linguagem muito eficiente, rápida e estável;
- Possui fácil integração com banco de dados, pode-se citar: MySQL, PostgreSQL, SQL Server, Oracle, entre outros;

Com estas vantagens é possível demonstrar como o PHP é uma linguagem flexível, em que é possível escolher o sistema operacional onde o script será executado, escolher o banco de dados, programar utilizando uma linguagem estrutural ou orientada a objetos, e até mesmo utilizar ambas em conjunto. Tais pontos positivos que fazem o PHP a linguagem de programação Web mais difundida de todos os tempos, no que equivale a 59% em relação às outras linguagens utilizadas.

2.2.2 HTML

O HTML (*HyperText Markup Language*) é uma linguagem de marcação, onde é possível definir a posição de elementos e separar o texto e informações separadamente. As

páginas em HTML serão identificadas pelos *Browsers*, nos quais tem como a principal função interpretar, exibir e formatar as páginas Web.

Segundo Silva (2011), desde a invenção da Web o HTML evoluiu por sete versões:

- HTML
- HTML +
- HTML 2.0
- HTML 3.0
- HTML 3.2
- HTML 4.0
- HTML 4.01
- HTML5 (versão atual)

Para Quierelli (2010) o HTML precisa de várias tags para auxiliar a formatação de elementos, que com o tempo foram desenvolvidas tecnologias para a elaboração de paginas para a internet, tanto para a elaboração de paginas mais leves quanto o desenvolvimento rápido e conteúdo mais dinâmico, dentre essas tecnologias estão as Folhas de Estilos, conhecidas como CSS e a linguagem de programação o PHP, onde essas ferramentas são fundamentais para criar conteúdo profissional para a internet, em que o CSS irá formatar o conteúdo das paginas, como cor de fonte, fundo da pagina e o PHP permitirá inserir o conteúdo dinamicamente nas paginas, através do acesso ao banco de dados onde se encontra as informações.

2.2.3 CSS

A necessidade de criar páginas mais bonitas e atraentes faz com que seja necessário aplicar estilos a linguagem de marcação. O CSS é a linguagem mais difundida nos dias atuais em questões de desenvolvimento de paginas Web e seu uso é indispensável para manter harmonia entre os elementos.

> As CSS surgiram como uma solução às deficiências e limitações que a linguagem HTML começou a apresentar já há algum tempo. Com o aparecimento de sites mais complexos, repletos de recursos e informações – levando em conta, sempre, o aspecto estético das páginas -, houve como consequência negativa a formação de

> arquivos HTML compostos por informações e formatações misturadas. Isso tornou a manutenção de tais arquivos um tanto complicada e demorada, o que, na maioria das vezes, tornava as páginas inconsistentes e sem padrão (MIYAGUSKU, 2007, pag. 8).

Segundo Jobstraibizer (2010), o CSS é utilizado para definir a apresentação de documentos escritos em uma linguagem de marcação, como o HTML ou XML, e o seu principal beneficio é a separação entre o formato e o conteúdo de um documento, que veio através do avanço da quantidade de paginas Web publicadas na internet e que veio com a intenção de torna-los mais leves, bonitos, limpos e dinâmicos e que tem se tornado uma das ferramentas mais amplamente difundida nos dias atuais.

Segundo Silva (2012), o CSS possui alguns níveis, assim classificados pela W3C, alegando ser o termo mais apropriado. Cada nível projetado é com base no anterior, e sempre possui o conjunto de funcionalidades do nível anterior, fazendo com que ele seja plenamente suportado. Dessa forma, um agente de usuário que suporte as funcionalidades das CSS de nível atual suporta também todas as funcionalidades das CSS de níveis anteriores.

2.2.4 JavaScript

O JavaScript é uma linguagem suportada por quase todos os browsers, e que por sua vez pode haver diferenças de sintaxes entre os mesmos, principalmente em navegadores de internet mais antigos ou desatualizados. Quando o JavaScript é embutida ao código HTML pode criar interações entre o usuário e o browser sem que haja necessidade de que o script execute dentro do servidor.

> JavaScript é uma linguagem de programação leve, interpretada e com recursos de orientação a objetos. O núcleo de uso geral da linguagem foi incorporado ao Netscape, Internet Explorer e em outros navegadores Web e aprimorado para programação Web com adição de objetos que representam a janela do navegador e seu conteúdo. Essa versão de JavaScript do lado cliente permite que o conteúdo executável seja incluído em páginas Web - significa que uma página Web não precisa mais de HTML estático, mas pode incluir programas que interagem com o usuário, controlam o navegador e criam conteúdo dinamicamente (FLANAGAN, 2002, p. 19).

A integração do JavaScript as páginas HTML é indispensável para que haja uma melhor interatividade entre o cliente e o browser, fazendo que assim o conteúdo seja gerado dinamicamente sem a necessidade de mandar requisições no servidor ou mesmo recarregar a página. O JavaScript ao contrário de uma linguagem compilada como o Java, ou interpretada como o PHP, é pouco evoluída, pois não permite nenhuma privacidade no que se refere aos códigos criados, ele pode ser visualizado logo após o carregamento da página pelo browser ao entrar em seu código fonte.

O comentário no código fonte de uma aplicação é fundamental para qualquer linguagem de programação, e com o JavaScript por sua vez não seria diferente, da mesma forma em que o desenvolvedor pode encontrar ou entender o script meses depois de desenvolver uma aplicação, ele também proporcionaria com que os visitantes curiosos possam compreender o próprio código fonte, coisa que nem sempre é desejado. Alguns desenvolvedores optam por tentar comprimir este código fonte, esse processo é chamado de *minified*, no qual tem como finalidade de dificultar o entendimento do código e diminuir o tamanho do arquivo em *bits* para que o arquivo seja carregado mais rapidamente pelo browser.

2.3 Sistemas de Gerenciamento de Banco de Dados (SGBD)

Para a construção de um software é necessário saber se ele ira armazenar algum tipo de registro que possa ser utilizado posteriormente, fazendo com que esses dados sejam armazenados em algum meio físico e que possua segurança, pois seus dados poderão conter informações que não poderão ser acessados por qualquer pessoa.

> Um banco de dados é um conjunto de dados armazenados em um computador. Esses dados, se observados separadamente, não têm valor nenhum, mas quando utilizados em ordem, revelam informações que poderão ser usadas futuramente, por isso um banco de dados deve ser seguro e nunca ficar exposto a pessoas não autorizadas, (LOBO, 2008, pag. 7).

Esse conjunto de dados que são armazenados poderá ser utilizado a modo que auxilie o armazenamento de informações para um melhor desempenho de um software, um banco de

dados pode ter informações confidenciais por esse motivo que ele tem que ser seguro e se possível nunca ser acessado por pessoas ou sistemas não autorizados.

Segundo Date (2004), um Sistema Gerenciador de Banco de Dados (SGBD) é um sistema computadorizado de manutenção de registros, onde os usuários podem realizar diversas operações envolvendo os dados armazenados, tais como:

- Acrescentar novos arquivos ao banco de dados;
- Inserir dados em arquivos existentes;
- Buscar dados de arquivos existentes;
- Excluir dados de arquivos existentes;
- Alterar dados em arquivos existentes;
- Remover arquivos existentes do banco de dados.

Podem-se citar alguns exemplos de SGBD's, tais como:

- MySQL;
- PostgreSQL;
- Firebird;
- mSQL.

Ambos SGBD's executam comandos na linguagem SQL (*Structured Query Language*) na qual representa um padrão mundial de manipulação de dados.

2.3.1 MySQL

O MySQL é um SGBD *open source* mais popular no desenvolvimento de sistemas Web, principalmente por possuir um bom desempenho, velocidade e facilidade de uso em sistemas deste tipo. Como o PHP o MySQL também é multiplataforma, ou seja, roda em diversos sistemas operacionais, tais como: Windows, Linux, Mac OS, entre outros.

Segundo Savoia (2010), podem ser citadas algumas vantagens ao utilizar o MySQL com o PHP:

- O PHP e o MySQL funcionam muito bem juntos, isso porque foram desenvolvidos tendo em mente um ao outro;
- Ambas as linguagens são *open source power*, que significa que são gratuitos para uso;

- Ambos têm suporte do centro acadêmico que constantemente está favorecendo com melhorias e atualizações;
- Como foi desenvolvido de forma simplificada, o funcionamento dos ambos é extremamente rápido;
- Uma das principais vantagens das duas linguagens, é que para começar a programar em PHP com MySQL não é necessário conhecer tudo sobre a linguagem.

O MySQL utiliza a Linguagem Query Estruturada, conhecida como SQL, a qual é usada para inserir, recuperar e manipular dados.

> A arquitetura do MySQL é muito diferente da dos outros servidores de banco de dados e é útil para uma grande variedade de objetivos. MySQL não é perfeito, mas é flexível o suficiente para trabalhar bem em ambientes muito exigentes, como aplicações web. Ao mesmo tempo, MySQL pode potencializar aplicações embutidas, depósitos de dados, indexação de conteúdo e software de distribuições, sistemas redundantes altamente disponíveis, processamento de transação on-line (OLTP), e muito mais, (SCHWARTZ et al., 2009, pag. 1).

O MySQL além de poder ser trabalhada em ambientes que sejam bastante exigentes, possui também uma facilidade de uso que possibilita aos usuários uma boa interação com a ferramenta, sendo essas funcionalidades que a tornaram um dos bancos de dados mais utilizados no mundo, 9 entre 10 websites no mundo utilizam o MySQL, onde podem ser citados: Facebook, Youtube, Twitter, entre outros.

2.4 Técnicas e Padrões de Desenvolvimento Web.

Com a evolução da web, ouve-se a necessidade de criar algumas técnicas e padrões de desenvolvimento, tanto com as linguagens de programação, como o PHP, e com as linguagens de marcação e folhas de estilos, como o HTML e o CSS.

2.4.1 World Wide Web Consortium (W3C)

No mês de outubro de 1994, Tim Berners-Lee em parceria com a CERN, onde a web foi por ele inventada, criaram o *World Wide Web Consortium* (W3C), com sede no Laboratório da Ciência da Computação do Massachusetts Institute of Technology (MIT).

O W3C é uma organização com a finalidade de criar padrões e normas de desenvolvimento para criação e interpretação de conteúdos para a internet.

2.4.2 Orientação a Objetos (O.O)

O reaproveitamento de código é um passo importante no desenvolvimento de um sistema, pois ele pode possibilitar um melhor aproveitamento de recursos e funcionalidades de um sistema.

> Similar ao conceito de eletrônica, o paradigma da orientação a objetos proporciona as bases para a modularização, reutilização e simplificação, tanto das tarefas de programação quanto da depuração de erros (depuração – localização e correção dos erros). Assim, como a eletrônica agrupou e encapsulou centenas e milhares de componentes discretos em um único componente integrado, a ideia da orientação a objetos é agrupar e encapsular dados e rotinas, enfim, programas inteiros dentro de objetos. Modularizando e simplificando a programação, o paradigma da orientação a objetos fornece aos programadores uma maneira de administrar melhor as milhares de linhas de códigos de seus projetos. Através da orientação a objetos, mesmo os mais arrojados, sofisticados e complexos programas podem ser vistos, numa última instância, com a conexão de alguns blocos, ou melhor, objetos (BRUNO; ESTEROZI; BATISTA NETO, 2010, pag 103).

Segundo Oglio (2007) a Orientação a Objetos é um paradigma que representa toda a filosofia da construção de um sistema. Que ao invés de construir um sistema que sejam agrupados de acordo com o contexto, como em linguagens procedurais citadas pelo autor como Cobol, Clipper, Pascal, a orientação trabalha com uma óptica mais próxima do mundo real, onde são trabalhados com objetos que são mais bem compreendidas e mais conhecidas no dia-a-dia.

2.4.3 Model-View-Controller (MVC)

O MVC (*Model-View-Controller*) é um padrão de arquitetura de software que separa a interface em que o usuário interage das suas regras de negócio. O MVC pode ser considerado um *design pattern*, pois é possível criar uma solução reutilizável em que poderá servir de base para diversas aplicações.

> Quando MVC é referenciado como um *design pattern* ou padrão de design de software, isso ocorre porque MVC vem sendo adotado como solução recorrente para um problema conhecido, uma solução reutilizável. Essa possibilidade de utilizar um mesmo modelo de desenvolvimento para diversos problemas distintos acabou fazendo com que muitas pessoas apontassem MVC como um *design pattern*. Um framework de certo modo também tem essa característica por servir de base para vários projetos distintos. Você não precisa escrever uma classe de conexão a dados para cada novo website ou sistema que cria, porque isso já está escrito no framework. Assim, basta reutilizar a solução já desenvolvida, testada e conhecida (GABARDO, 2012, pag. 18).

2.5 Frameworks

Um framework é uma biblioteca orientada a objetos que reúne funcionalidades comuns em diversas aplicações, tornando-se uma aplicação reutilizável e de base para diversos projetos.

A figura a seguir pode demonstrar um exemplo de estrutura de um framework e quando pode ser aplicado:

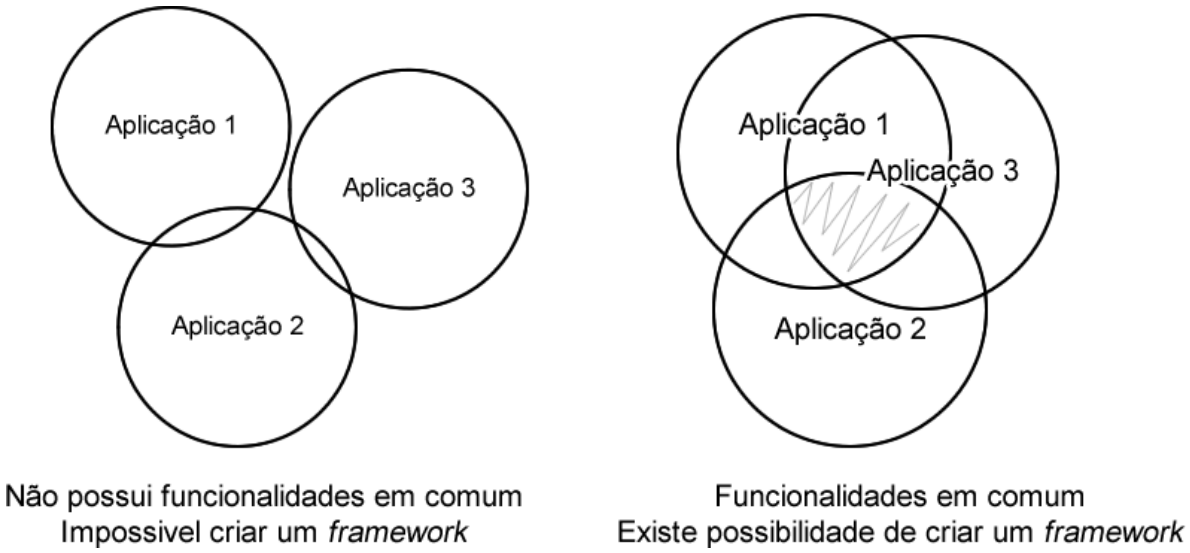

Figura 1 - Aplicação de um framework
Fonte: Acervo do Autor

Com a utilização de *frameworks* é possível criar padrões de desenvolvimento, tanto na parte de desenvolvimento do layout, quanto no comportamento dos elementos e estrutura de códigos. Existem diversos *frameworks* que auxiliam no desenvolvimento de um software, podem ser citados como exemplos de *frameworks* em PHP o *CodeIgniter, Zend Framework, Cake PHP* e no CSS o *Twitter Bootstrap* e *Initializr*. A utilização de um *framework* facilita o controle de um erro ou atualização, pois o mesmo irá afetar em todos os projetos da mesma forma.

2.5.1 CodeIngiter

Segundo Jobstraibizer (2009) o *CodeIgniter* é um *framework* bastante difundido na comunidade PHP, é extremamente leve, pois possui pouquíssimas e pequenas bibliotecas, dentre suas principais vantagens se destacam:

- Utilização da metodologia MVC;
- Possibilidade de utilização das URL's amigáveis;
- Extensível por meios de bibliotecas, plug-ins ou casses adicionais;
- Não requer gerador de *templates*, podendo ser utilizado com blocos de HTML puros;
- Suporte a *ActiveRecord Database.*

O *CodeIgniter* é um *framework* mantido pela *EllisLab*, sua última versão é a 2.1.4, dando suporte ao PHP 4.x e PHP 5.x. Por ser uma ferramenta *open source* e de simples utilização, proporciona aos desenvolvedores recursos bem completos para criação de aplicações web.

Pode-se considerar o *CodeIgniter* como um *design pattern* ou padrão de design de software, pois o mesmo pode ser visto como uma solução reutilizável em que é utilizada como base de diversos projetos de software.

2.5.2 Twitter Bootstrap

O Twitter Bootstrap (TB) é um framework que nasceu das mãos dos desenvolvedores @mdo e @fat. Inicialmente tratava-se de um esforço para documentar determinados padrões de design e UI (*user interface*) dentro do Twitter, mas a coisa tomou proporções maiores e seus autores decidiram disponibilizar o Twitter Bootstrap no Github se tornando um dos mais populares por lá.

O TB pode ser visto como um conjunto de HTML, CSS e JavaScript que proporciona aos usuários de forma mais rápida, eficiente e profissional planejar e desenvolver o front-end de websites, sistemas e produtos web através de seus elementos de UI e interações (http://desenvolvimentoparaweb.com/miscelanea/twitter-bootstrap/).

Algumas vantagens de utilizar o TB são:

- Sistemas de grids
- Layouts fixos e fluidos.
- Classes pré-definidas para elementos HTML (botões, formulários, fontes, entre outros).
- Elementos JavaScript (modais, abas, tooltips, entre outros).

III. SISTEMAS DE INFORMAÇÃO

3.1 Sistema de informação

Um sistema de informação é um aglomerado de elementos que pode ser composto por dados, pessoas, materiais e informações em geral. Estes elementos se relacionam tornando-se um todo unificado para a divulgação de sua informação de forma precisa.

Todos nós interagimos diariamente com sistemas de informação, usamos os caixas automáticos dos bancos, os scanners de leitura de preços dos supermercados que identificam nossas compras usando o código de barras, e, ainda, obtemos informação em quiosques por meio de telas sensíveis ao toque.

Um Sistema de informação é composto de um subsistema social e de um subsistema automatizado. O primeiro abrange os processos, documento, informações e pessoas. O segundo fundamenta-se dos meios automatizados que se interligam com os elementos do subsistema social.

O número de subsistemas varia de acordo com o que é necessário e depende do ponto de vista de cada pessoa ou de seu objetivo.

O papel de sistema de informação é conectar dois mundos: A Tecnologia da informação e a Organização. No meio deles existe o sistema de informação, que repassa tudo o que um lado precisa saber sobre o outro.

É através deste sistema que será determinado o que deve ser coletado da organização para armazenar na parte de tecnologia e o que deverá ser devolvido para a parte organizacional. Sem os sistemas de informação, de nada serve a tecnologia, pois não saberiam o que coletar nem o deverá ser devolvido.

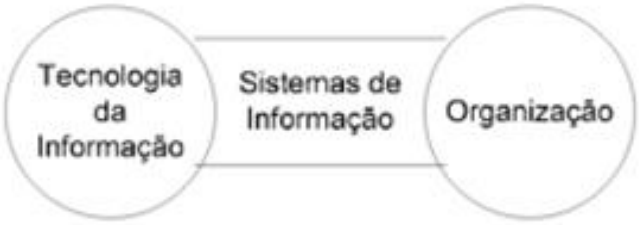

Figura 2 - Demonstração da ponte entre Tecnologia e Organização.
Fonte: Senac.

Matsuda (2007), diz que: "sistemas de informação são processos administrativos que envolvem processos menores que interagem entre si. O sistema é dividido em subsistemas que podem ser: produção/serviço venda distribuição, materiais, financeiro, recursos humanos e outros, dependendo do tipo de empresa. O departamento de informática da empresa cruza esses subsistemas, o que leva a uma abordagem sistemática integrativa, envolvendo questões de planejamento estratégico da empresa".

É responsabilidade do Analista de Sistemas: designar os objetivos do sistema de informação, as informações que este irá gerenciar, as pessoas que farão parte e os processos.

3.2 Classificando Sistemas de Informação

Os sistemas de informação podem ser classificados de maneiras diferentes. Vários tipos de sistemas de informação, por exemplo, podem ser classificados conceitualmente ora como operações, ora como sistema de informação gerencial.

Eles são classificados desta maneira para destacar os papéis principais que cada um desempenha nas operações e administração de um negócio.

As classificações mais conhecidas são as seguintes:

Tabela 1 - Classificação sistemas de informação
Fonte: Próprio Autor

Sistemas de informação de gestão: para solucionar problemas empresariais em geral;
Sistemas de informação executiva para diretores, gerentes e administradores;

Sistemas de informação estratégicos ou de apoio à decisão: analisam as distintas variáveis de negócio para a tomada de decisões;
Sistemas de informação operacional ou de processamento de transações: que gerem a informação referente às transações que têm lugar numa empresa;
Sistemas especializados que emulam o comportamento de um especialista numa área concreta.
Sistemas de automatização de escritórios aplicações que ajudam no trabalho administrativo;

3.3 Sistemas de informação na saúde

A primeira aplicação prática da computação relevante para a área da saúde foi o desenvolvimento de um sistema de processamento de dados baseado em cartões perfurados, criado por Herman Hollerith em 1890. Primeiramente utilizado para a realização do censo dos Estados Unidos daquele ano, o sistema foi, logo a seguir, adotado para solucionar problemas nas áreas de epidemiologia e saúde pública (BLOIS & SHORTLIFFE, 1990).

Um Sistema de Informação Hospitalar, veterinário ou não, (SIH) pode ser definido como um sistema computadorizado, podendo ser web ou desktop, desenvolvido para facilitar o gerenciamento de toda a informação administrativa e assistencial de um hospital.

Mesmo que exista uma série de SIHs vindas do exterior, alguns no mercado nacional, ainda são poucos os que conseguem se adequar às reais necessidades de suporte à saúde em um hospital. Não por sua estrutura ser um software mal desenvolvido, mas pela falta de abrangência que é necessária nesse ramo.

3.4 Sistemas de informação em hospitais veterinários

Com o surgimento do microcomputador na década de 70, a informática sofreu um notável processo de democratização e de popularização. Com eles, surgiram também as linguagens de alto nível – bastante próximas da linguagem coloquial, que, associadas a poderosos sistemas operacionais, provocaram mudança radical no perfil dos usuários de

computador. Os "sacerdotes" de Harrison, "ao invés de lidar com analfabetos em computação, enfrentavam, agora, um grande número de usuários finais que tinham conhecimentos básicos de informática, liam revistas especializadas, compravam computadores para seus filhos usarem em casa e já não ficavam impressionados diante de alguém que comentasse sobre ROM e RAM" (HARRISON, 2006).

O impacto dessa nova tecnologia na prática da medicina é surpreendente. As técnicas não invasivas de produção de imagem, como a ultra-sonografia, a medicina nuclear, a tomografia e a ressonância magnética, alteraram sensivelmente o processo de diagnóstico médico (DAMASCO, 1999).

Matsuda (2007) diz que: Os sistemas de informação em saúde podem monitorar o processo de assistência à saúde e aumentar a qualidade da assistência ao paciente por auxiliar no processo de diagnóstico ou na prescrição da terapia, por permitir a inclusão de lembretes clínicos para o acompanhamento da assistência, de avisos sobre interações de drogas, de alertas sobre tratamentos duvidosos e desvios dos protocolos clínicos.

Em hospitais veterinários, os softwares mais conhecidos e utilizados são: o SISVET e o VetSoft.

3.4.1 SISVET

O SISVET é desenvolvido por AMZ Comercial Assessoria e Consultoria em Sistemas Específicos Ltda. Inicialmente, a AMZ se dedicava única e exclusivamente à criação de softwares sem similares ou específicos para atender empresas, adequando assim, os sistemas às necessidades da empresa.

Atualmente, a AMZ, criadora do SISVET, é a única empresa formada por profissionais de informática voltada única e exclusivamente a este segmento de mercado, dedicando-se a comercialização e implantação do SISVET em estabelecimentos veterinários, oferecendo treinamento do pessoal, informatização do negócio, construção de redes de computadores, home-pages e suporte permanente.

Um dos Diretores-Proprietário da AMZ, Michel Zimberknopf, também se dedica à assessoria e consultoria na área pet/vet, auxiliando na organização e estruturação de clínicas e pet/shops, lucratividade e gerenciamento administrativo e de pessoal, além de assessoria para abertura de novas empresas. (SISVET, 2000)

Não apenas em hospitais veterinários, como também pet shops, o SISVET é conhecido por ter uma interface de fácil manuseio, assim não é preciso ter algum curso adicional em informática para utilizá-lo.

O SISVET possui um sistema de demonstração, onde se pode fazer download do sistema, com todas as suas funcionalidades, mas com validade de apenas dois ou três dias. Dessa maneira, o futuro cliente poderá saber como o software funciona e se existe nele todas as funcionalidades que necessita antes de implantar em um hospital veterinário.

Figura 3 - Tela inicial

Fonte: SISVET

Figura 4 - Ficha do Animal

Fonte: SISVET

3.4.2 VetSoft

O VetSoft age entre os melhores sistemas informatizados para gestão de clínicas veterinárias e pet shops do paí, sendo utilizado em pequenas, médias, grandes clínicas e hospitais em praticamente todos os estados brasileiros.

Apesar de robusto em suas funcionalidades, é um software de aprendizado fácil e rápido por possuir interface amigável e de fácil entendimento.

O VetSoft pode ser implantado de duas formas: somente VetSoft (responsável pelo gerenciamento simples) ou VetSoft + Hiper (sistema que emite os cupões fiscais).

No VetSoft (Gerencial, simples) é utilizado para gestão da sua clínica ou petshop. É nele que são gerenciadas informações de animais e proprietários, agendamentos, vacinas e etc. Não há integração para emissão de cupom fiscal.

Já o VetSoft + Hiper (Gerencial + Cupom Fiscal) é uma integração entre o sistema VetSoft e o sistema Hiper, da empresa Uni4 Sistemas. Com esta integração é possível vender com emissão de cupom fiscal.

Assim como o SISVET, o VetSoft também possui um sistema de demonstração. Basta realizar o download e a instalação que o software se apresentará. Para aqueles que não quiserem realizar o download da demonstração do software, o VetSoft possui uma página em seu site que apresenta as telas do software em uso.

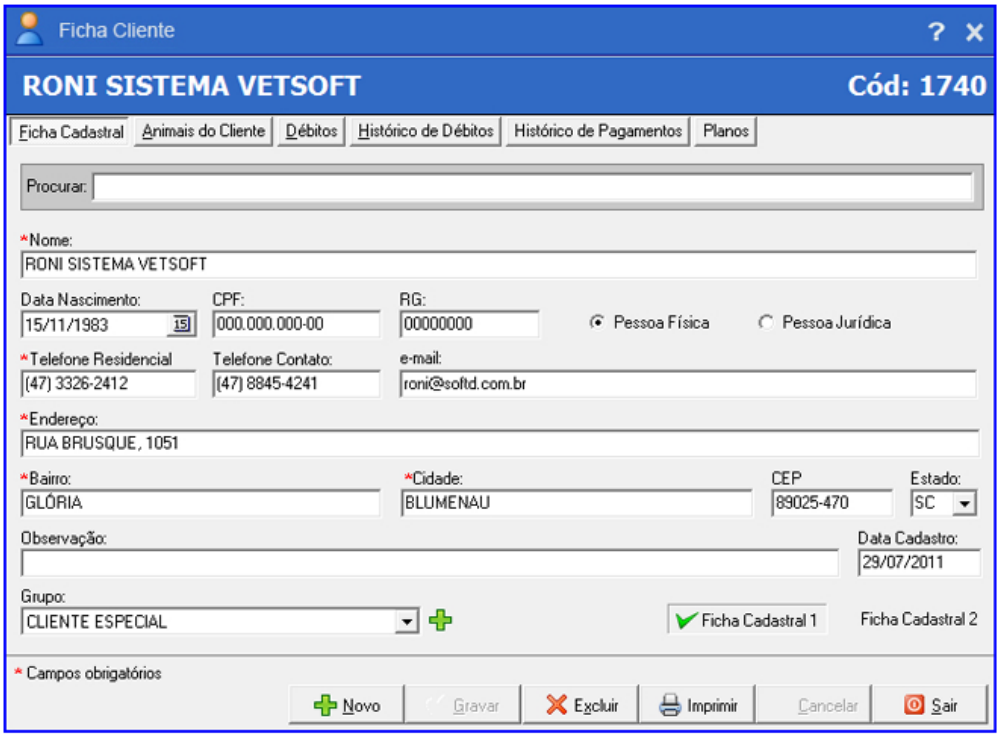

Figura 5 – Demonstração Ficha do cliente

Fonte: VetSoft

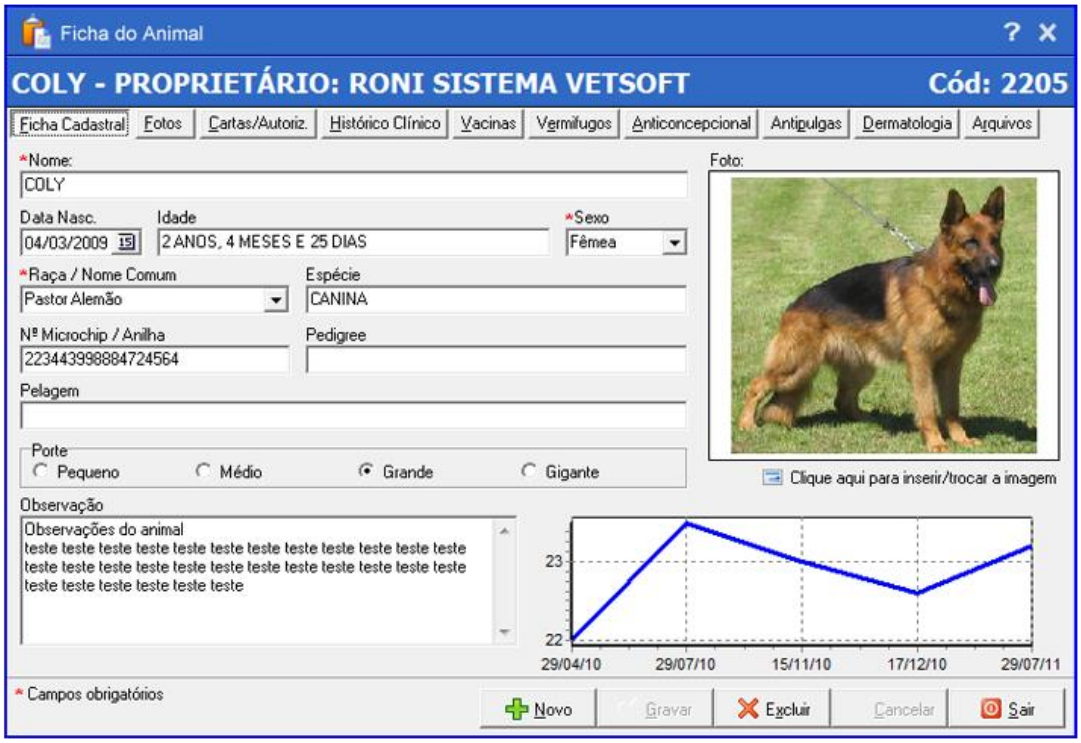

Figura 6 – Demonstração Ficha do animal

Fonte: VetSoft

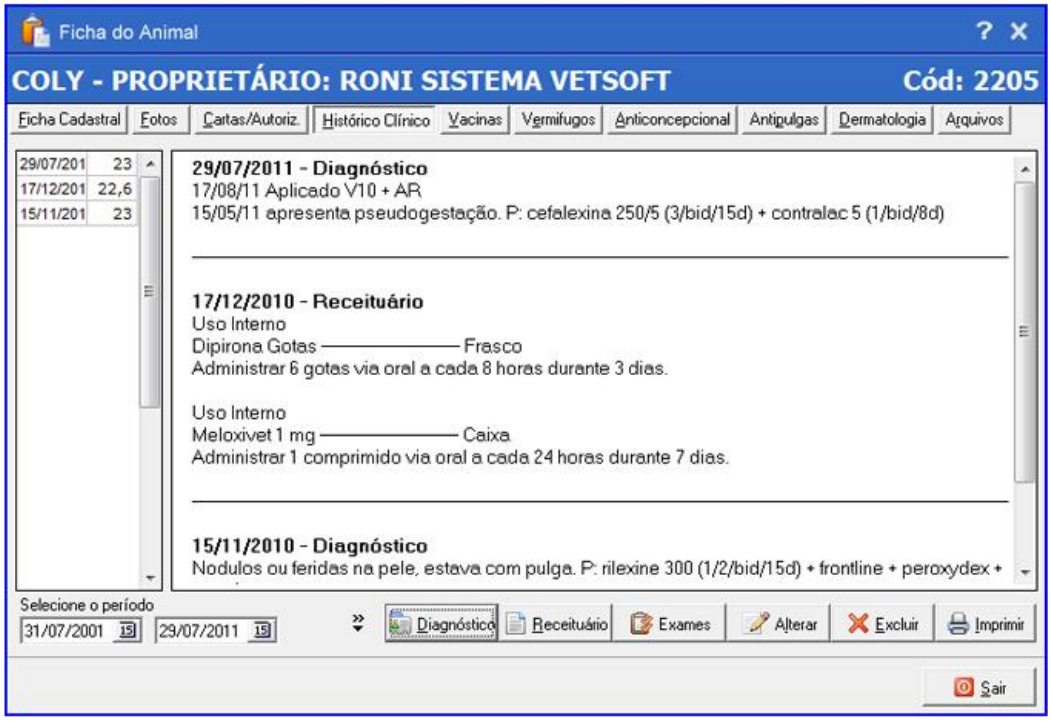

Figura 7– Demonstração Ficha do Animal- Histórico hospitalar

Fonte: VetSoft

IV. PROJETO

Para que seja realizado o desenvolvimento de um sistema computacional é necessário realizar um estudo sobre as funcionalidades necessárias para a implementação do sistema. Com base no levantamento destes dados e pesquisas bibliográficas foi possível abstrais as necessidades e problemas encontrados atualmente no gerenciamento do hospital veterinário do Centro Universitário Unifacvest.

O UniVet foi desenvolvido na linguagem de programação PHP, uma linguagem *open source*, e que possui uma vasta comunidade na internet na qual pode-se basear no desenvolvimento de aplicações para internet, e que por sua vez também foi utilizado um *framework* denominado *CodeIgniter*, o qual possui sua estrutura de desenvolvimento em MVC que possibilita um melhor aproveitamento de recursos e tempo de desenvolvimento.

4.3 UML

A Unified Modeling Language (UML) é uma linguagem de modelagem de sistemas, em que auxilia visualizar seu desenho e a comunicação entre objetos.

Com a utilização da UML é possível permitir que os desenvolvedores visualizem seus produtos e projetos em diagramas padronizados.

4.2 Diagrama de classes

O diagrama de classes a seguir possui a representação das classes necessárias para o desenvolvimento do UniVet.

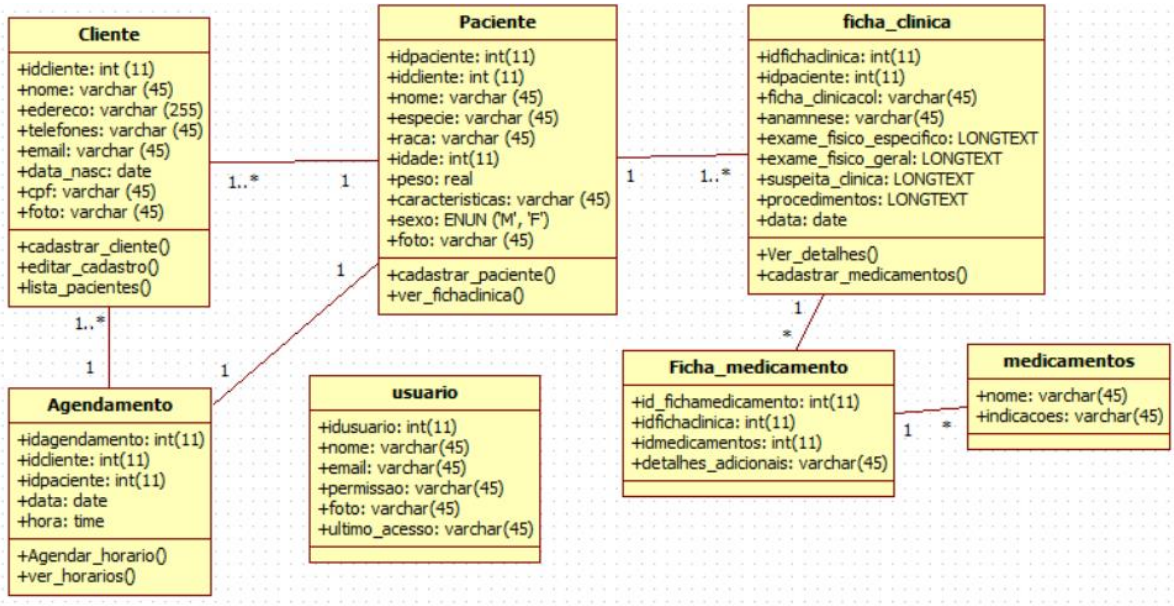

Figura 8 – Diagrama de Classes

Fonte: Próprio Autor

4.3 Caso de uso

O caso de uso a seguir mostra de maneira mais clara a utilização do sistema através do caso de uso.

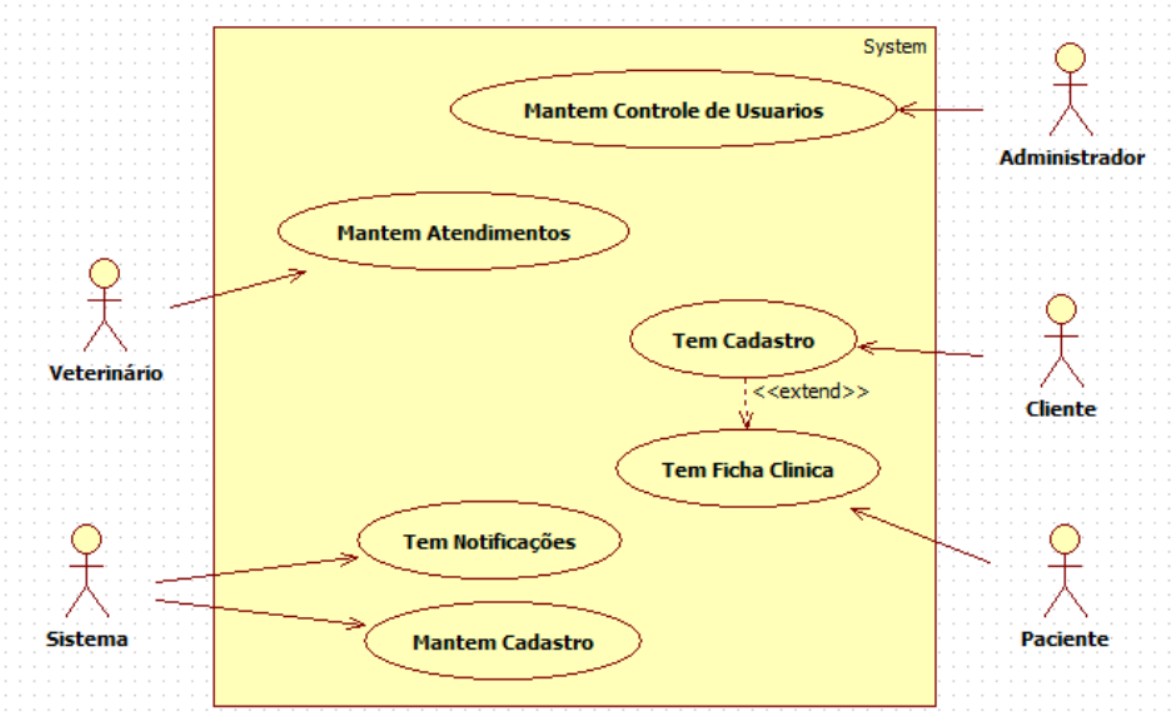

Figura 9 – Diagrama de Caso de Uso

Fonte: Próprio Autor

4.3.1 Descrição curso normal e curso alternativo dos casos de uso

Curso Normal

1. Cadastra Cliente
2. Sistema verifica se Cliente já está cadastrado
3. Cliente é cadastrado pelo Veterinário
4. Sistema emite “Cliente Cadastrado”

Curso Alternativo

1. Cadastra Cliente
2. Sistema Verifica se Cliente já está cadastrado

2.1 O sistema emite “Cliente já cadastrado”

2.2 Abandonar Use Case

Curso Normal

1. Cadastra Paciente
2. Sistema Verifica se Paciente já está cadastrado
3. Paciente é cadastrado pelo Veterinário.
4. Sistema emite Paciente Cadastrado.

Curso Alternativo

1. Cadastra Paciente
2. Sistema Verifica se Paciente já está cadastrado

2.1 O sistema emite “Paciente já cadastrado”

2.2 Abandonar Use Case

Curso Normal

1. Cadastra Ficha Clinica
2. Sistema verifica dados informados.
3. Ficha clinica é cadastrada.

4. Sistema emite ”Ficha Clinica Cadastrada”.

Curso Alternativo

1. Cadastra Ficha Clinica
2. Sistema verifica dados informados.

2.1 Dados informados incorretos.

2.2 Sistema emite “Dados Informados estão incorretos”.

2.3 Abandonar Use Case

Curso Normal

1. Consulta Ficha Clinica
2. Sistema verifica filtros
3. Sistema retorna dados da ficha clinica
4. Sistema emite a “Ficha Clinica”

Curso Alternativo

1. Consulta Ficha Clinica
2. Sistema verifica filtros

2.1 Sistema não encontra dados da ficha clinica

2.2 Sistema emite “Nenhuma ficha clinica cadastrada”

2.3 Abandonar Use Case

Curso Normal

1. Verificar agendamentos
2. Verifica próximas consultas agendadas
3. Envia para usuário via e-mail notificação de agendamento
4. Retorna “notificação enviada”

Curso Alternativo

1. Verificar agendamentos
2. Verifica próximas consultas agendadas

2.1 Nenhuma consulta agendada

2.2 Retorna “Nenhuma consulta agendada”

2.3 Abandonar Use Case

Curso Normal

1. Verifica na agenda
2. Verifica hora e data do agendamento
3. Verifica se o Veterinário que irá atender está disponível
4. Sistema retorna “Horário agendado”

Curso Alternativo

1. Verifica na agenda
2. Verifica hora e data do agendamento
3. Verifica se o Veterinário que irá atender está disponível

3.1 O veterinário não tem horário disponível

3.2 Sistema emite “Horário não disponível, tente outro horário”

3.3 Abandonar Use Case

4.4 Diagrama de sequência

O diagrama de sequência a seguir irá demonstrar a troca de mensagens existentes entre os objetos, sendo destacados os métodos mais relevantes.

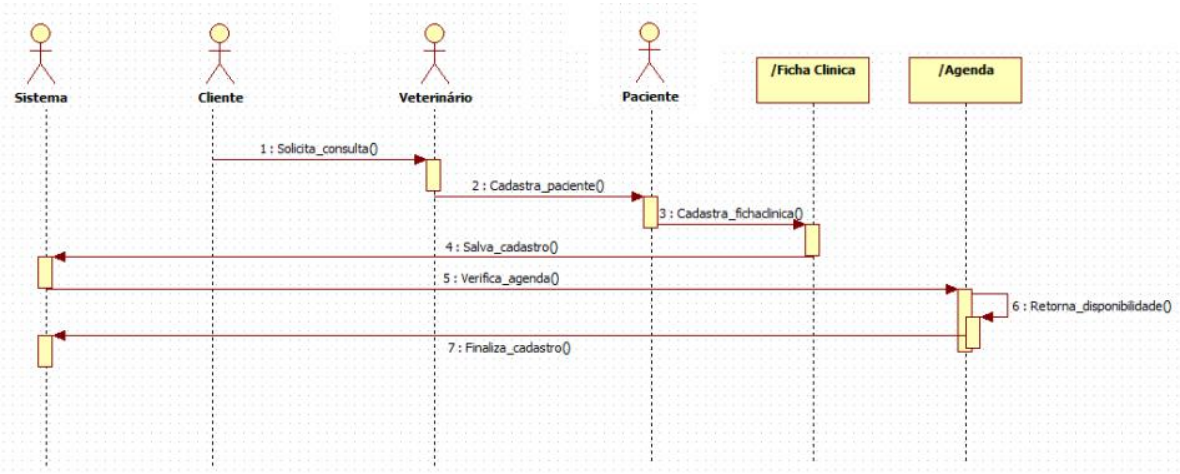

Figura 10 - Diagrama de sequência

Fonte: Próprio Autor

4.5 Diagrama de atividades

A seguir será demostrado o diagrama de atividades, onde pode ser visualizado os passos para a utilização do sistema.

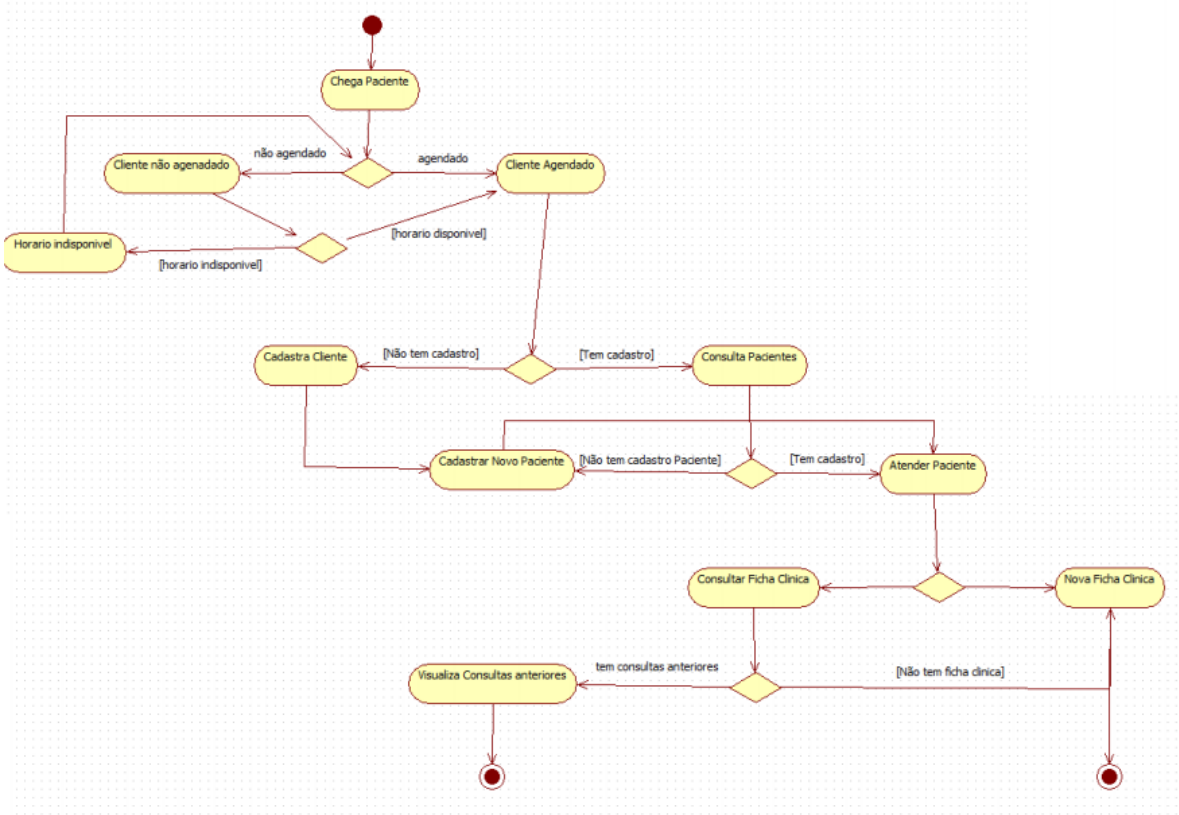

Figura 11 - Diagrama de atividades

Fonte: Próprio Autor

4.6 Interfaces do UniVet

A seguir serão apresentadas algumas das interfaces do UniVet.

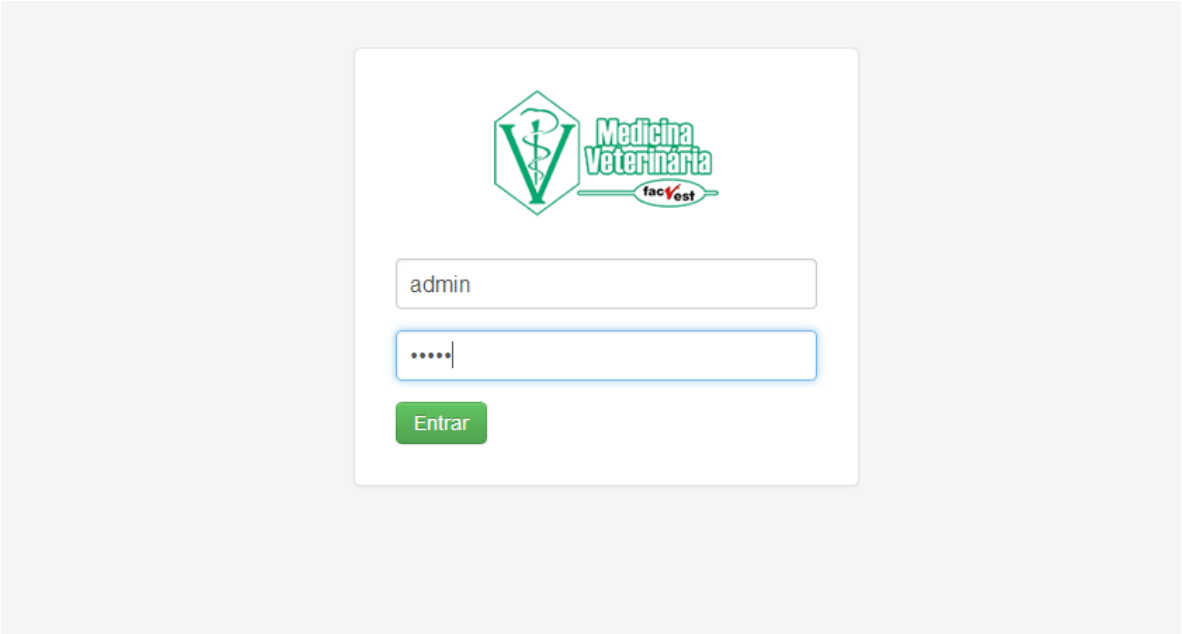

Figura 12 - Tela de Login

Fonte: Próprio Autor

A figura 10 apresenta a tela inicial do sistema, onde o usuário deverá entrar com o login e senha para a utilização do sistema.

Figura 13 - Tela de apresentação após login

Fonte: Próprio Autor

A figura 11 apresenta a tela inicial do sistema após o login, onde o usuário terá acesso às funcionalidades do sistema.

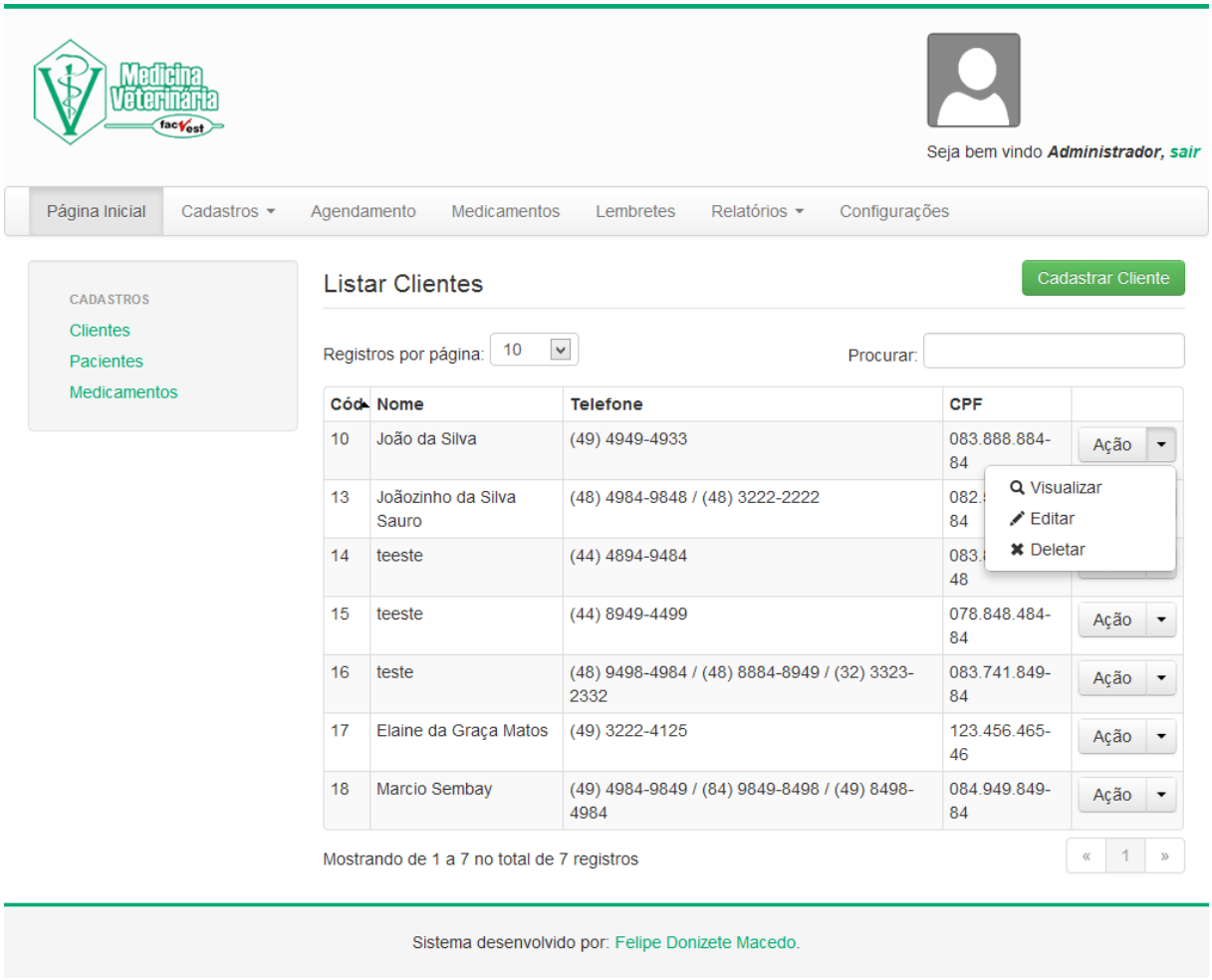

Figura 14 - Listagem de clientes

Fonte: Próprio Autor

A figura 12 apresenta a tela de listagem de clientes, onde é possível visualizar os cadastros, alterar, apagar e incluir novo cadastro.

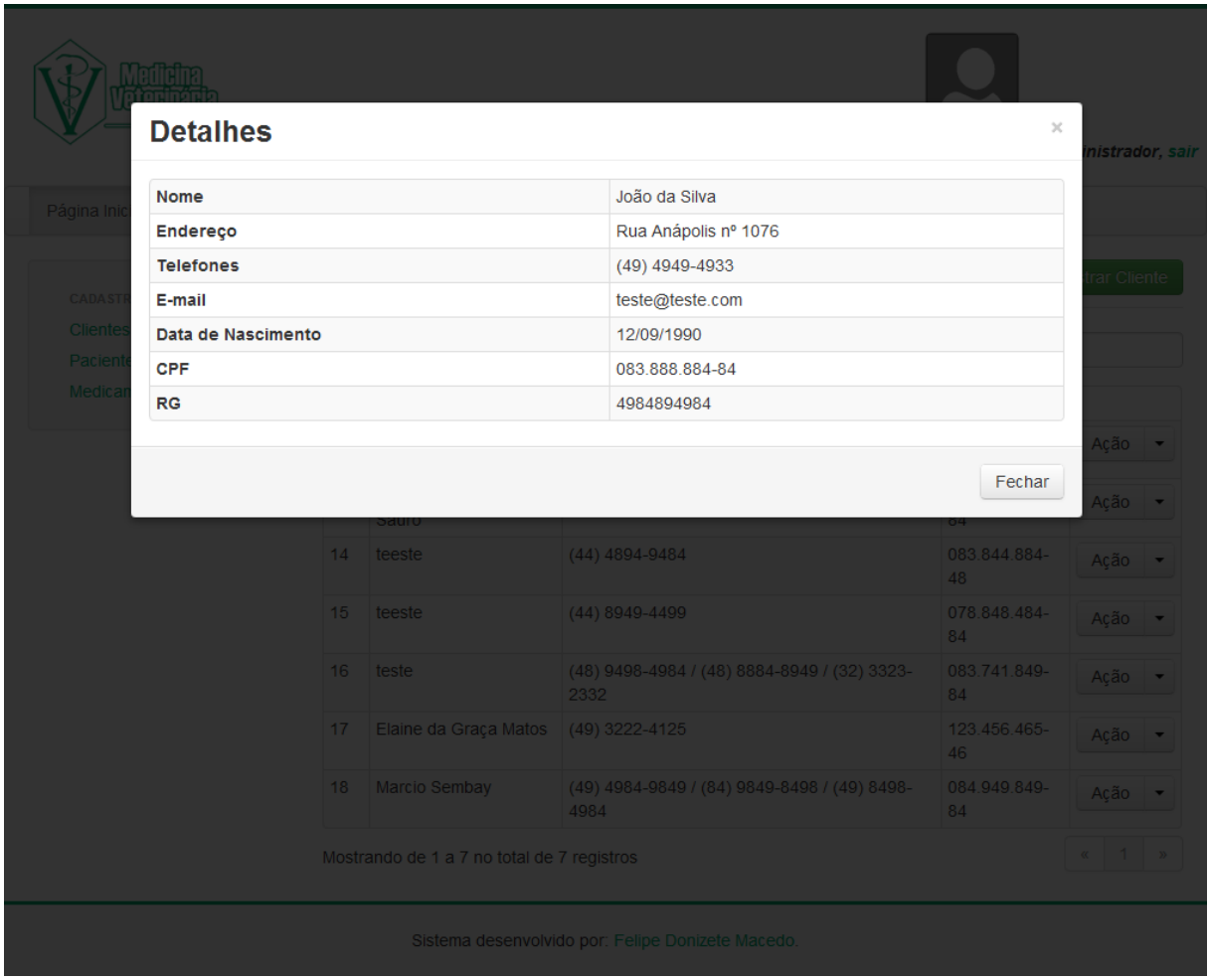

Figura 15 - Detalhes do cliente na página de cadastros

Fonte: Próprio Autor

A figura 13 apresenta os detalhes do cliente ao clicar em visualizar cadastro.

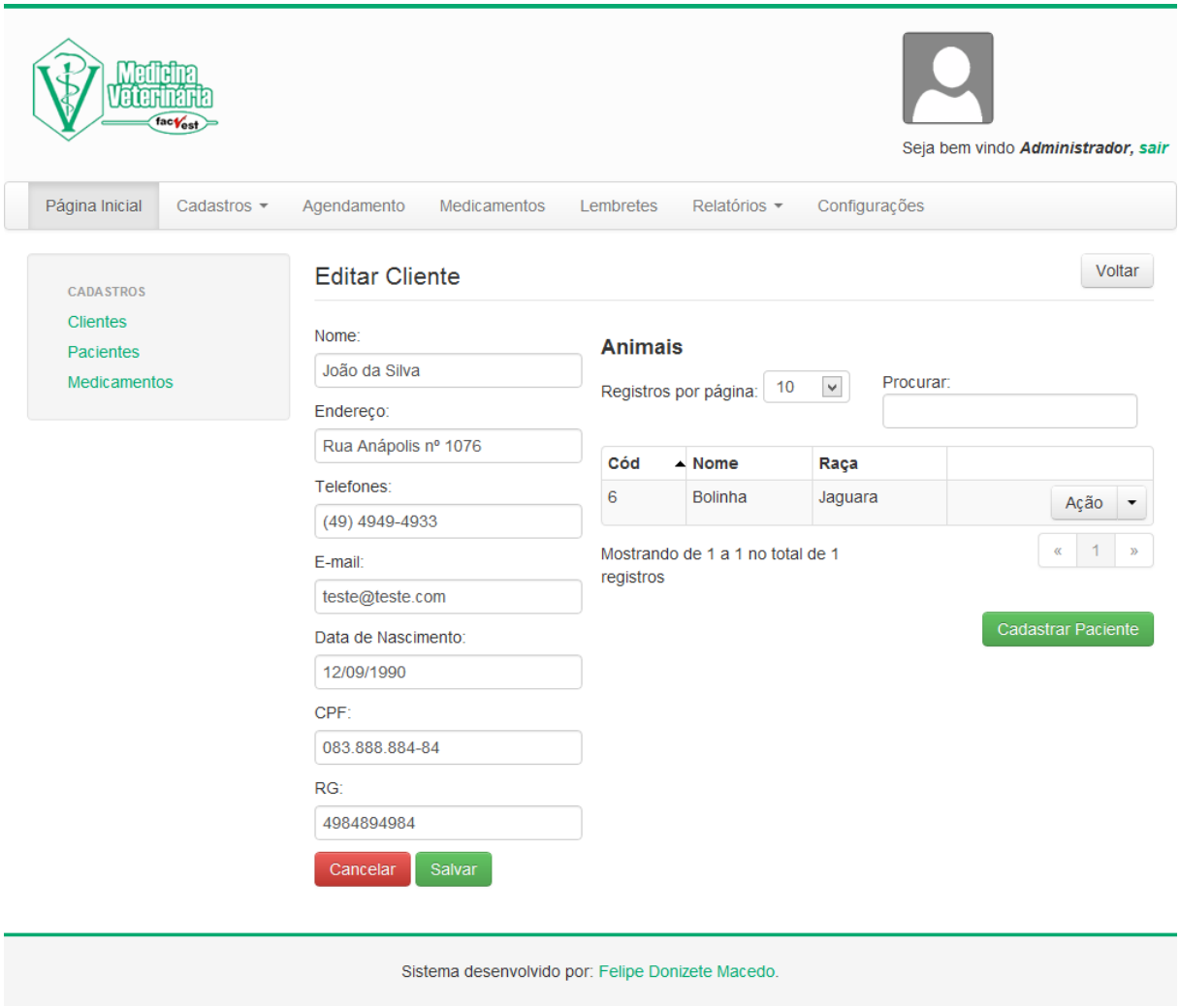

Figura 16 - Editar cadastro do cliente

Fonte: Próprio Autor

A figura 14 apresenta o formulário de edição do cadastro do cliente, onde também é possível visualizar os pacientes relacionados ao cadastro.

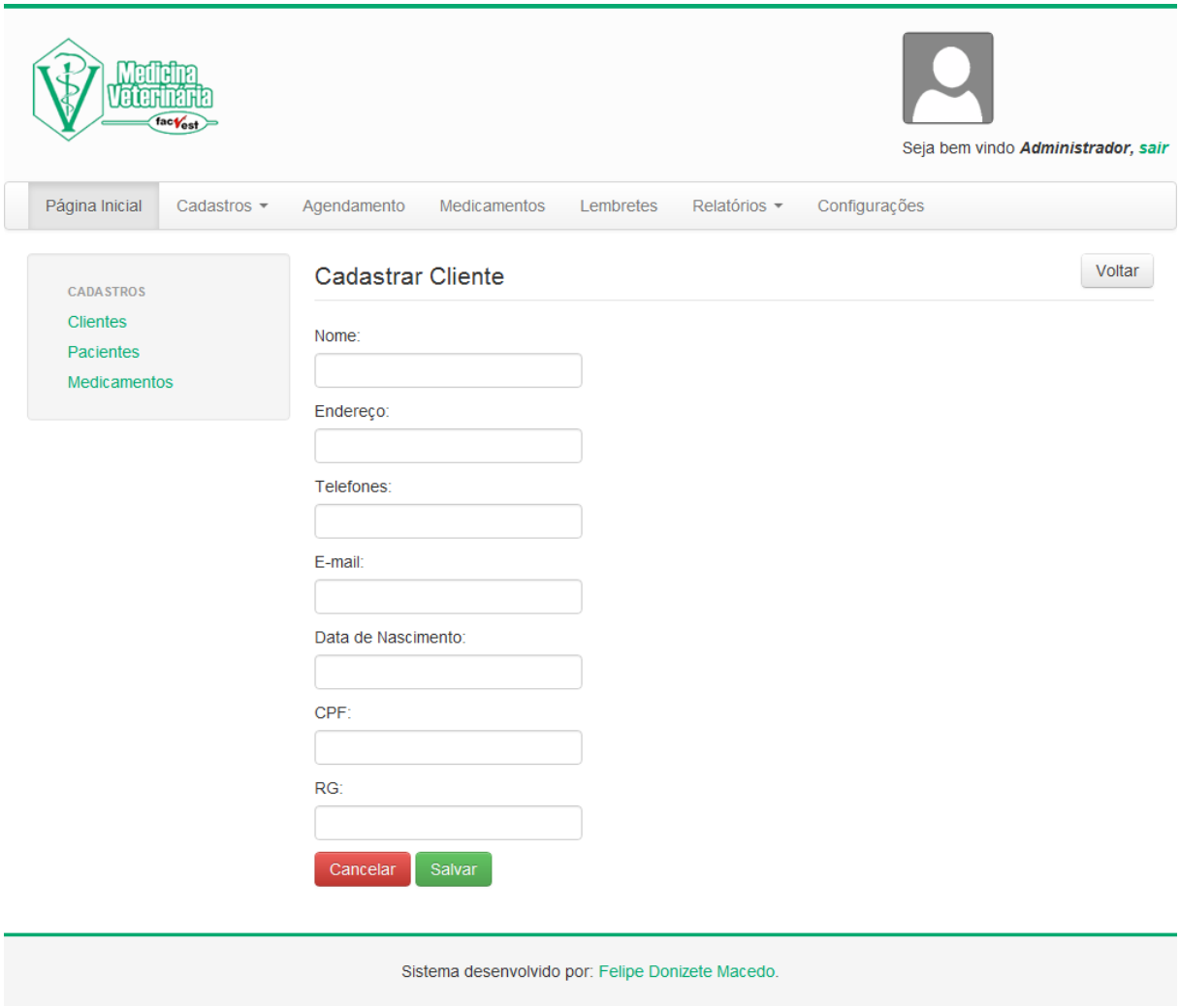

Figura 17 - Tela de inserção de novo cadastro

Fonte: Próprio Autor

A figura 15 apresenta o formulário para cadastro de um novo cliente ao sistema.

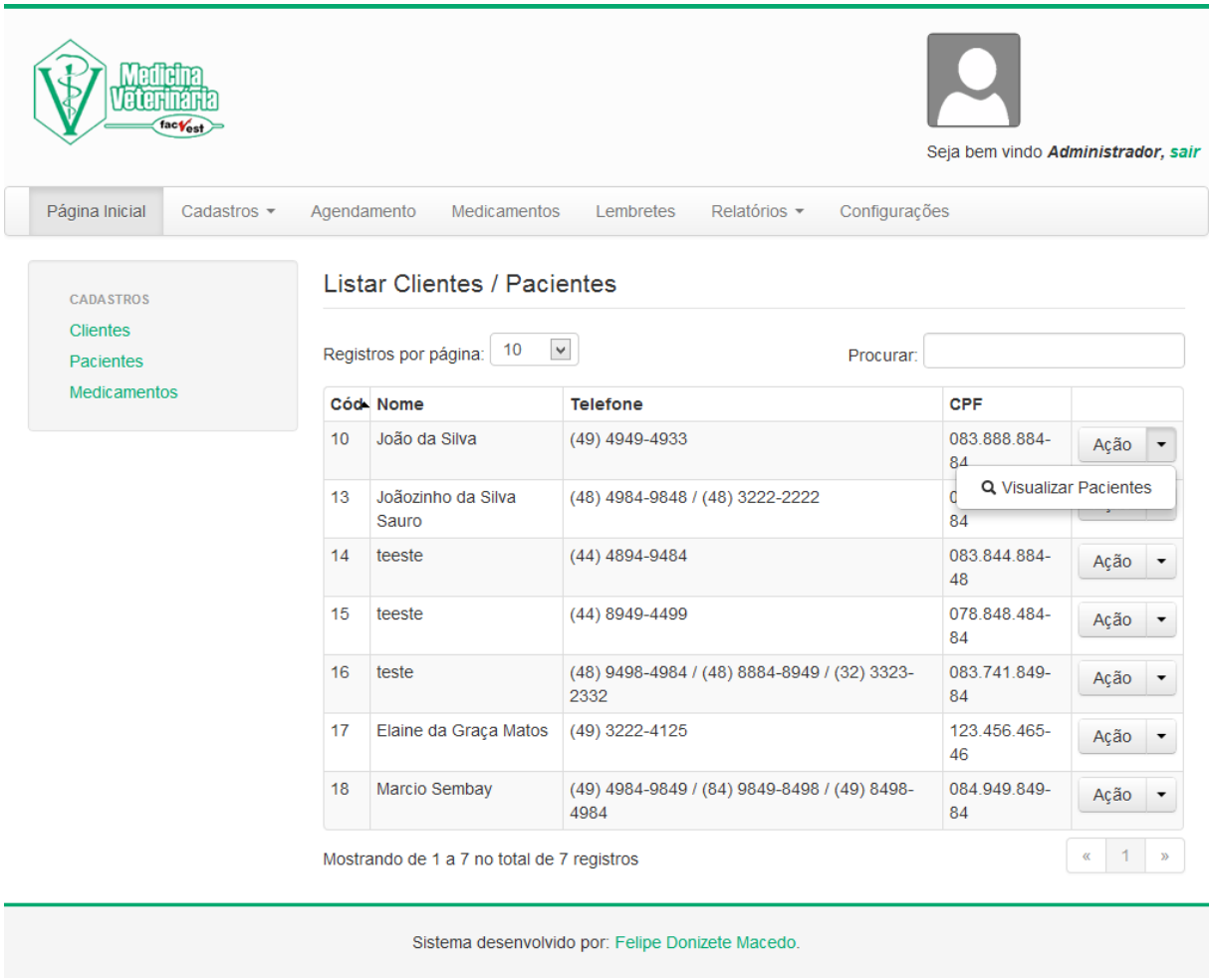

Figura 18 - Tela de listagem de pacientes por cliente

Fonte: Próprio Autor

A figura 16 apresenta a listagem dos clientes com a finalidade de visualizar os pacientes relacionados aos mesmos.

Ao clicar em visualizar pacientes mostrado na figura anterior apresenta o seguinte resultado.

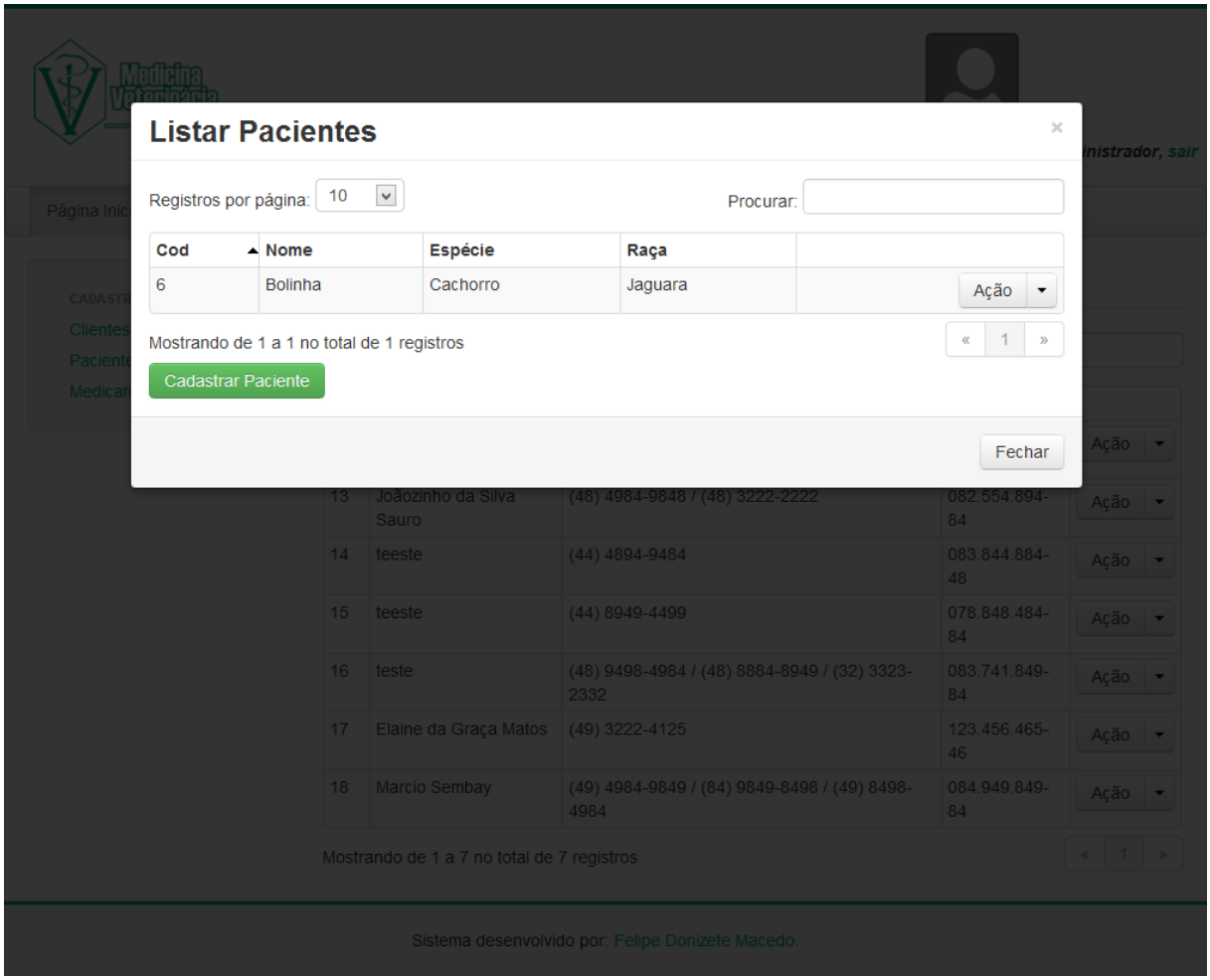

Figura 19 - Listar Pacientes

Fonte: Próprio Autor

A figura 17 apresenta a tela de listagem dos pacientes, onde é possível cadastrar, deletar, inserir novo paciente e visualizar a ficha clinica.

Figura 20 - Ficha clínica do paciente

Fonte: Próprio Autor

A figura 18 apresenta a tela de ficha clinica do paciente, onde é possível visualizar o histórico, vacinas e imprimir autorizações.

Ao clicar no símbolo “+” ao lado do proprietário abrirá uma tela igual a Figura 13 com as informações do cliente.

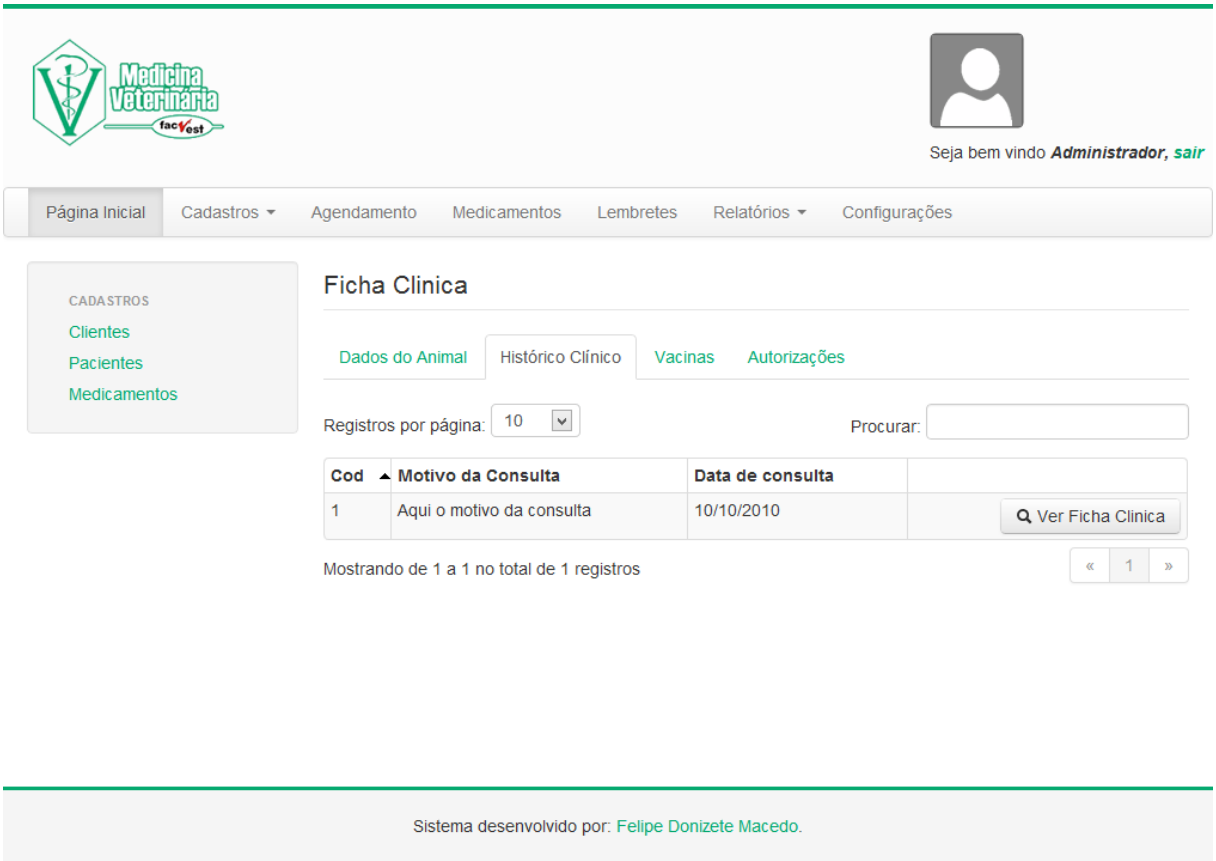

Figura 21 - Histórico Clínico

Fonte: Próprio Autor

A figura 19 apresenta a tela de histórico do paciente, onde é possível ver a ficha clinica do mesmo por consulta.

Ao clicar em ver ficha clinica abre uma tela com as informações da consulta e opção para impressão.

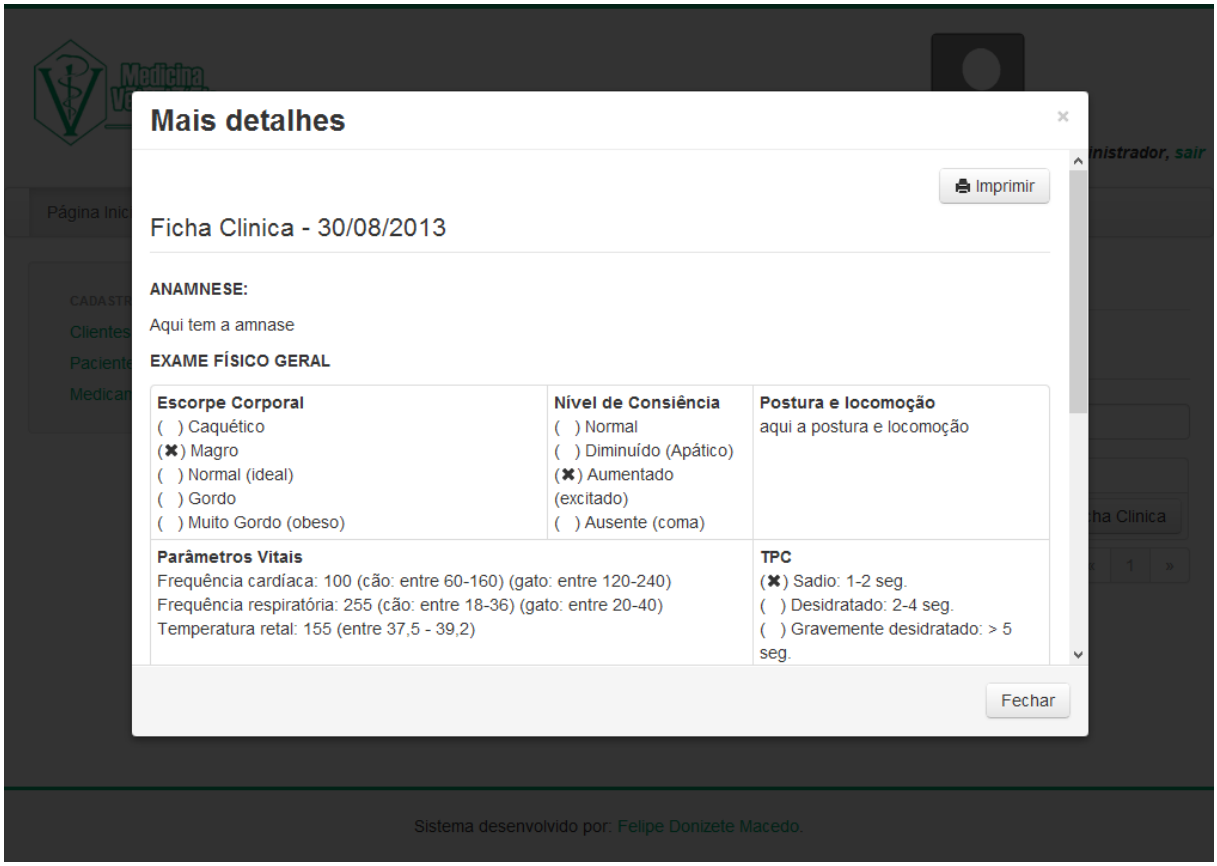

Figura 22 - Ficha clínica especifica

Fonte: Próprio Autor

A figura 20 apresenta a tela de ficha clinica por data, onde é possível visualizar o atendimento completo do paciente, e possibilitar a impressão.

A impressão deste formulário será apresentado abaixo, a qual aparece quando clicamos em Imprimir.

Clínica Médica de Pequenos Animais

Paciente: Bolinha **Espécie:** 1 **Raça:** Jaguara **Data Nascimento:** 10/10/2010

Ficha Clinica - 30/08/2013

ANAMNESE:

Aqui tem a amnase

EXAME FÍSICO GERAL

Escorpe Corporal ( ) Caquético (X) Magro ( ) Normal (ideal) ( ) Gordo ( ) Muito Gordo (obeso)	**Nível de Consiência** ( ) Normal ( ) Diminuído (Apático) (X) Aumentado (excitado) ( ) Ausente (coma)	**Postura e locomoção** aqui a postura e locomoção
Parâmetros Vitais Frequência cardíaca: 100 (cão: entre 60-160) (gato: entre 120-240) Frequência respiratória: 255 (cão: entre 18-36) (gato: entre 20-40) Temperatura retal: 155 (entre 37,5 - 39,2)		**TPC** (X) Sadio: 1-2 seg. ( ) Desidratado: 2-4 seg. ( ) Gravemente desidratado: > 5 seg.
Mucosas (X) Normocoradas (rósea) ( ) Hipocoradas (pálida) ( ) Hipercoradas, congesta, hiperêmica (vermelha) ( ) Ictérias (amarela) ( ) Cianótica (azulada)	**Elasticidade da Pele** ( ) Redução de elasticidade da pele discreta ou sem alteração (X) Redução da elasticidade da pele (de 2 a 4 s) ( ) Elasticidade da pele bem diminuída (de 6 a 10 s) ( ) Marcante perda da elasticidade da pele (> 10 s)	
Linfonodos Aqui os linfonodos		

EXAME FÍSICO ESPECÍFICO:

Aqui você escreve o exame específico

SUSPEITA(S) CLÍNICA(S):

Aqui cadastra as suspeitas clinicas

PROCEDIMENTOS:

Aqui você cadastra os procedimentos

Figura 23 - Ficha clínica para impressão

Fonte: Próprio Autor

A figura 21 apresenta a tela de impressão da ficha clínica do cliente, onde é possível visualizar as mesmas informações da figura 20, mas com a formatação correta para impressão.

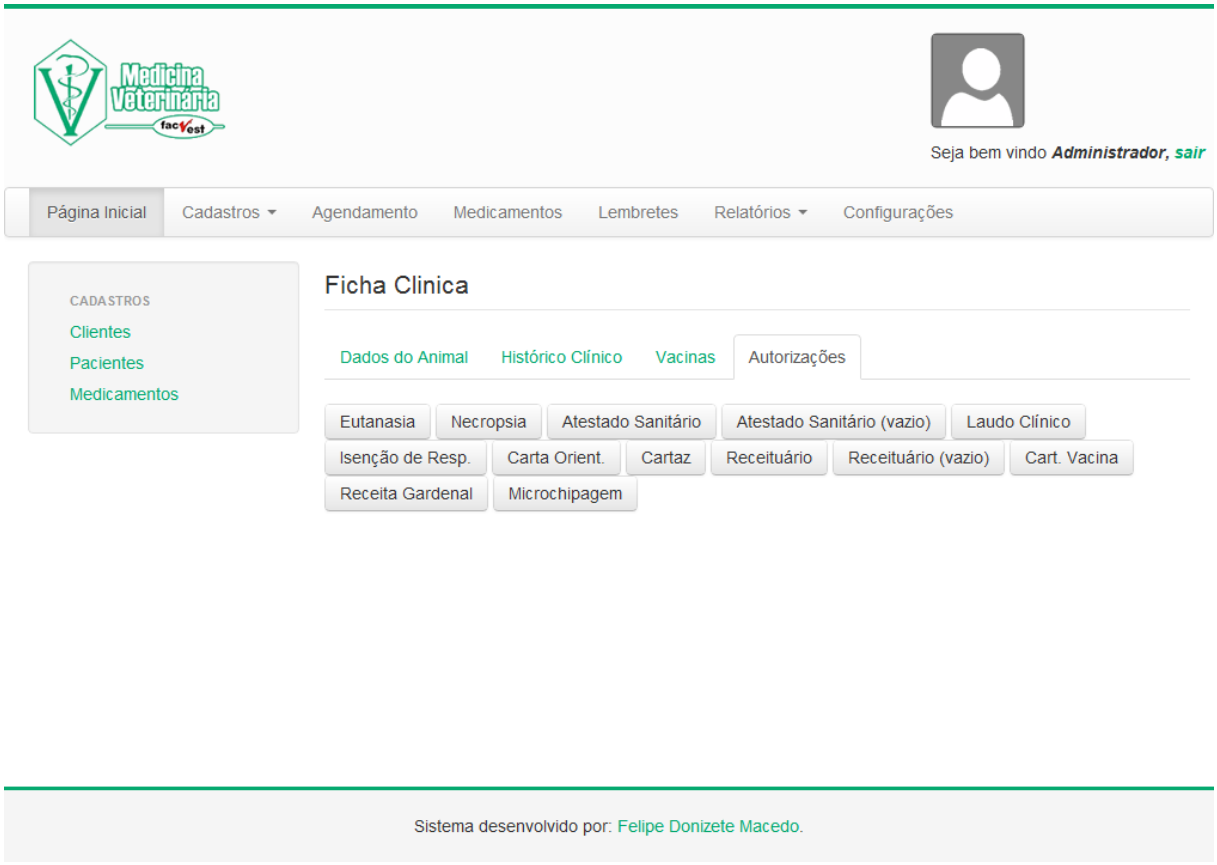

Figura 24 - Tela de autorizações

Fonte: Próprio Autor

A figura 22 apresenta a tela de autorizações, onde irá gerar os arquivos para impressão e assinatura dos clientes.

.

V. CONSIDERAÇÕES FINAIS

No estudo e desenvolvimento deste projeto foi observado que o hospital veterinário Unifacvest assim como os demais, necessita um sistema informatizado para oferecer um melhor serviço a seus clientes.

Quando bem planejado, a automação de processos traz inúmeros benefícios para os administradores, como em qualquer estabelecimento comercial. Baseado nesse ponto, o UniVet foi desenvolvido para que os usuários possam administrar as consultas, atendimentos, e fazer o acompanhamento dos pacientes de uma forma mais eficaz.

A linguagem de programação PHP é uma linguagem eficiente de desenvolvimento em aplicações Web, com inúmeras vantagens, pode-se citar entre elas: rapidez, estabilidade, fácil integração com banco de dados, e *open source.*

O UniVet apresenta alguns diferenciais, principalmente que ele foi desenvolvido em linguagem Web, que oferece suporte a multiplataforma, podendo ser acessado por diversos sistemas operacionais existentes no mercado através de um navegador de internet, apresentando uma interface limpa e de fácil acesso para os usuários.

VI. REFERÊNCIAS BIBLIOGRÁFICAS

ARAYA, Elizabeth Roxana Mass; VIDOTTI, Silvana Aparecida Borsetti Gregorio. **Criação, Proteção e uso legal da informação em ambientes da Word Wide Web.** Unesp, 2010.

BRUNO, Odemir M. et al. **Programando para internet com PHP**. Rio e Janeiro: Brasport, 2010.

BRUNO, Odemir M.; ESTEROZI, Leandro F.; BATISTA NETO, João E. S.. **Programando para a internet com PHP.** São Paulo: Brasport Livros e Multimidia Ltda, 2010.

CHICOLI, Milton. **Guia prático de criação de sites.** São Paulo: Digerati Books, 2008.

COMER, Douglas E. **Redes de Computadores E Internet 4 Ed**. São Paulo: Bookman, 2007.

DATE, C. J. **Introdução a Sistemas de Banco de Dados**. Rio de Janeiro: Campus, 2004.

ENGHOLM JUNIOR, Hélio. **Engenharia de Software na Prática.** São Paulo: Novatec, 2010.

FLANAGAN, David. **JavaScript:** Guia Definitivo. São Paulo: Novatec, 2002.

GABARDO, Ademir Cristiano. **PHP e MVC com Codeingiter.** São Paulo: Novatec, 2012.

GAMMA, Erich et al. **Padrões de Projetos:** soluções reutilizáveis de software orientado a objetos. Porto Alegre: Bookman, 2000.

GRIFFITHS, Adam. **CodeIgniter 1.7:** Professional Development. Birminghan: Packt Publishing, 2010.

HARRISON, Jeffrey S.. Trad. Luciana de Oliveira da Rocha. **Administração Estratégica de Recursos e Relacionamentos.** Porto Alegre: Editora Bookman, 2006.

JOBSTRAIBIZER, Flávia. **Criação de Sites com CSS**. São Paulo: Digerati, 2009.

LOBO, Edson Junio Rodrigues. **Curso prático de MySQL**. São Paulo: Digerati, 2008.

LOUDON, Kyle. **Desenvolvimento de Grandes Aplicações Web.** São Paulo: Novatec, 2010.

MACINTYRE, Peter B.. **O melhor do PHP**. Rio de Janeiro: Alta Books, 2010.

MARTÍN, Alicia Ramos; MARTÍN, Mª Jesús Ramos. **APLICACIONES WEB**. Madrid: Paraninfo, 2011.

MIYAGUSKU, Renata Hiromi Minami. **Desvendando os recursos do CSS**. São Paulo: Digerati, 2007.

MUTO, Claudio Adonai. **PHP & MySQL Guia Introdutório.** Rio de Janeiro: Brasport, 2006.

NIEDERAUER, Juliano. **PHP para quem conhece PHP.** São Paulo: Novatec, 2008.

NIEDERAUER, Juliano. **Desenvolvendo Websites com PHP.** São Paulo: Novatec, 2011.

OGLIO, Pablo Dall'. **PHP-GTK**: Criando Aplicações Gráficas com PHP. São Paulo: Novatec, 2007.

O'BRIEN, James A. **Sistemas de informações e as decisões gerenciais na era da internet**. 3ª Ed. São Paulo: Editora Saraiva, 2010.

QUIERELLI, Davi Antonio. **Criando sites com HTML-CSS-PHP**: Construindo um projeto - Iniciante: Clube de Autores, 2012.

REZENDE, Denis Alcides. **Engenharia de Software e Sistemas de Informação.** Rio de Janeiro: Brasport, 2006.

SAMPAIO, Cleuton. **Web 2.0 e Mashups:** Reinventando a Internet. Rio de Janeiro: Brasport, 2007.

SAVOIA, Hugo Rossetti. **XHTML e CSS + PHP e MySQL:** Primeiros Passos. São Paulo: Ield, 2010.

SANDERS, William. **Padrões de Projeto em PHP.** São Paulo: Novatec, 2013.

SILVA, Maurício Samy. **HTML5:** A Linguagem que revolucionou a Web. São Paulo: Novatec, 2011.

SILVA, Maurício Samy. **CSS3:** Desenvolva aplicações web profissionais com uso dos poderosos recursos de estilização das css3. São Paulo: Novatec, 2012.

SCHWARTZ, Baron et al. **Alto Desempenho em MySQL**. Rio de Janeiro: Alta Books, 2009.

DAMASCO, Miguel. **Conceitos de Sistemas de Informação**. Disponível em: <http://www.profdamasco.site.br.com/SlidesFundamentosSI.pdf>. Acesso em: 05 novembro. 2013

JOBSTRAIBIZER, Flávia. **Guia profissional PHP.** São Paulo: Digerati Books, 2009.

Twitter Bootstrap. Disponível em: <http://desenvolvimentoparaweb.com/miscelanea/twitter-bootstrap/> Acesso em: 02 nov. 2013

MATSUDA. **Teoria dos sistemas**. Disponível em: <http://sites.mpc.com.br/gberaldo/Teoria%20dos%20sistemas.pdf>. Acesso em: 10 de novembro. 2013.

SIGULEM, Daniel. **Um Novo Paradigma de Aprendizado na Prática Médica da UNIFESP/EPM**. São Paulo, 1997. 177p./ Tese (Livre–Docência) – Universidade Federal de São Paulo – Escola Paulista de Medicina/.

PROFESSOR ANDRÉ. **Aulas Senac**. Disponível em: < http://187.7.106.14/andre/ads/sist-inf/2013-1/aulas/aula1/apostila%20SI-Senac.pdf>. Acesso em: 10 de novembro. 2013.

SISVET. Disponível em: <http://www.sisvet.com.br>. Acesso em: 10 de novembro de 2013.

VETSOFT Software Veterinário. Disponível em: <http://www.softwareveterinario.com.br/>. Acesso em: 10 de novembro de 2013.

VI. ANEXOS

```
<?php

if (!defined('BASEPATH'))
   exit('No direct script access allowed');

class Cadastros extends Core_Seguro {

   public $_nivel = 'cadastros';

   public function __construct() {
      parent::__construct();

      $menu_esq = $this->smarty->fetch('blocos/menu_esq.tpl');
      $this->smarty->assign('menu_esq', $menu_esq);
   }

   // função default do controller
   public function index() {
      $content = $this->smarty->fetch('cadastros/inicio.tpl');
      $this->smarty->assign('content', $content);

      $this->smarty->view('layout');
   }

   // função para listar clientes
   public function lista_clientes() {
      $this->checa_permissao('visualizar');

      $consulta_clientes = $this->model_banco->buscar_tudo_sort('clientes', 'id', 'asc');
      $this->smarty->assign('clientes', $consulta_clientes);

      $content = $this->smarty->fetch('cadastros/lista_clientes.tpl');
      $this->smarty->assign('content', $content);

      $this->smarty->view('layout');
   }

   // função para visualizar cliente
   public function visualiza_cliente($id) {
      $this->checa_permissao('visualizar');

      $consulta_clientes = $this->model_banco->buscar_tudo("clientes where id = '$id'");
      $this->smarty->assign('cliente', $consulta_clientes);

      echo $this->smarty->fetch('cadastros/visualiza_cliente.tpl');
   }

   // função para deletar cliente
   public function deleta_cliente($id) {
      $this->checa_permissao('excluir');

      $consulta_clientes = $this->model_banco->deletar("clientes", 'id', $id);
```

```
        $this->smarty->assign('sucesso', '<div class="alert alert-success">Cliente deletado com
sucesso!</div>');
        $this->lista_clientes();
    }

    // função para cadastrar cliente
    public function cadastra_cliente() {
        $this->checa_permissao('cadastrar');

        $config = array(
            array(
                'field' => 'nome',
                'label' => 'Nome',
                'rules' => 'required|min_length[3]'
            ),
            array(
                'field' => 'endereco',
                'label' => 'Endereço',
                'rules' => 'required'
            ),
            array(
                'field' => 'telefones',
                'label' => 'Telefones',
                'rules' => 'required'
            ),
            array(
                'field' => 'email',
                'label' => 'E-mail',
                'rules' => 'required|valid_email'
            ),
            array(
                'field' => 'data_nasc',
                'label' => 'Data de Nascimento',
                'rules' => 'required'
            ),
            array(
                'field' => 'CPF',
                'label' => 'CPF',
                'rules' => 'required'
            ),
            array(
                'field' => 'RG',
                'label' => 'RG',
                'rules' => 'required'
            )
        );

        $this->form_validation->set_rules($config);

        if ($this->form_validation->run() == TRUE) {
            $post = $this->input->post();
```

```
            $data_invertida = explode('/', $post['data_nasc']);

            $post['data_nasc'] = $data_invertida[2] . '-' . $data_invertida[1] . '-' . $data_invertida[0];

            $this->model_banco->db_insert('clientes', $post);

            $this->smarty->assign('sucesso', '<div class="alert alert-success">Cadastro gravado com
sucesso, clique <a href="cadastros/edita_cliente/' . $this->db->insert_id() . '">aqui</a> para inserir
pacientes!</div>');

            $this->lista_clientes();

            die();
        }

        $content = $this->smarty->fetch('cadastros/cadastrar_cliente.tpl');
        $this->smarty->assign('content', $content);

        $this->smarty->view('layout');
    }

    // função para editar cliente
    public function edita_cliente($id = '') {
        $this->checa_permissao('editar');
        $config = array(
            array(
                'field' => 'nome',
                'label' => 'Nome',
                'rules' => 'required|min_length[3]'
            ),
            array(
                'field' => 'endereco',
                'label' => 'Endereço',
                'rules' => 'required'
            ),
            array(
                'field' => 'telefones',
                'label' => 'Telefones',
                'rules' => 'required'
            ),
            array(
                'field' => 'email',
                'label' => 'E-mail',
                'rules' => 'required|valid_email'
            ),
            array(
                'field' => 'data_nasc',
                'label' => 'Data de Nascimento',
                'rules' => 'required'
            ),
            array(
                'field' => 'CPF',
```

```
            'label' => 'CPF',
            'rules' => 'required'
        ),
        array(
            'field' => 'RG',
            'label' => 'RG',
            'rules' => 'required'
        )
    );

    $this->form_validation->set_rules($config);

    if ($this->form_validation->run() == TRUE) {
        if ($id == '')
            $this->erro();

        $post = $this->input->post();
        $data_invertida = explode('/', $post['data_nasc']);

        $post['data_nasc'] = $data_invertida[2] . '-' . $data_invertida[1] . '-' . $data_invertida[0];

        $this->model_banco->updateTable('clientes', $post, 'id', $id);

        $this->smarty->assign('sucesso', '<div class="alert alert-success">Cadastro alterado com
sucesso!</div>');
    }

    $consulta_clientes = $this->model_banco->buscar_tudo("clientes where id = '$id'");
    $this->smarty->assign('cliente', $consulta_clientes);

    if (count($consulta_clientes) == 0)
        $this->erro();

    $consulta_pacientes = $this->model_banco->buscar_tudo("pacientes where idclientes = '$id'");
    $this->smarty->assign('pacientes', $consulta_pacientes);

    $this->smarty->assign('idclientes', $id);

    $content = $this->smarty->fetch('cadastros/edita_cliente.tpl');
    $this->smarty->assign('content', $content);

    $this->smarty->view('layout');
  }

}

<?php

if (!defined('BASEPATH'))
  exit('No direct script access allowed');

class Cadastros_pacientes extends Core_Seguro {
```

```
    public $_nivel = 'cadastros_pacientes';

    public function __construct() {
        parent::__construct();

        $menu_esq = $this->smarty->fetch('blocos/menu_esq.tpl');
        $this->smarty->assign('menu_esq', $menu_esq);
    }

    // função default do controller
    public function index() {
        $content = $this->smarty->fetch('cadastros_pacientes/inicio.tpl');
        $this->smarty->assign('content', $content);

        $this->smarty->view('layout');
    }

    // função para listar clientes
    public function lista_clientes() {
        $this->checa_permissao('visualizar');

        $consulta_clientes = $this->model_banco->buscar_tudo_sort('clientes', 'id', 'asc');
        $this->smarty->assign('clientes', $consulta_clientes);

        $content = $this->smarty->fetch('cadastros_pacientes/lista_clientes.tpl');
        $this->smarty->assign('content', $content);

        $this->smarty->view('layout');
    }

    // visualizar paciente
    function visualiza_paciente($id) {
        $this->checa_permissao('visualizar');

        $consulta_pacientes = $this->model_banco->sql("select p.*, e.nome as nome_especie from
pacientes p join especie e on p.especie = e.id where p.idclientes = '{$id}'");
        $this->smarty->assign('pacientes', $consulta_pacientes);

        $this->smarty->assign('idclientes', $id);

        echo $this->smarty->fetch('cadastros_pacientes/visualiza_paciente.tpl');
    }

    //cadastrar novo paciente
    function cadastra_paciente() {
        $this->checa_permissao('cadastrar');

        $config = array(
            array(
                'field' => 'idclientes',
                'label' => 'Você não pode fazer isso, favor recarregue a página',
```

```
            'rules' => 'required|integer'
        ),
        array(
            'field' => 'nome',
            'label' => 'Nome',
            'rules' => 'required|min_length[3]'
        ),
        array(
            'field' => 'especie',
            'label' => 'Espécie',
            'rules' => 'required'
        ),
        array(
            'field' => 'raca',
            'label' => 'Raça',
            'rules' => 'required'
        ),
        array(
            'field' => 'data_nasc',
            'label' => 'Data de Nascimento',
            'rules' => 'required'
        ),
        array(
            'field' => 'peso',
            'label' => 'Peso',
            'rules' => 'required'
        ),
        array(
            'field' => 'caracteristica',
            'label' => 'Caracteristica',
            'rules' => 'required'
        ),
        array(
            'field' => 'sexo',
            'label' => 'Sexo',
            'rules' => 'required'
        )
    );

    $this->form_validation->set_rules($config);

    if ($this->form_validation->run() == TRUE) {
        $post = $this->input->post();
        $data_invertida = explode('/', $post['data_nasc']);

        $post['data_nasc'] = $data_invertida[2] . '-' . $data_invertida[1] . '-' . $data_invertida[0];

        $this->model_banco->db_insert('pacientes', $post);

        $this->smarty->assign('sucesso', '<div class="alert alert-success">Cadastro gravado com
sucesso, clique <a href="cadastros_pacientes/ficha_animal/' . $this->db->insert_id() . '">aqui</a>
para ver a ficha clinica!</div>');
```

```
            // se cadastrar ele abre o lista clientes
            $this->lista_clientes();

            die();
        }

        $consulta_especies = $this->model_banco->buscar_tudo_sort('especie', 'nome', 'asc');
        $this->smarty->assign('especies', $consulta_especies);

        $this->smarty->assign('idclientes', $this->uri->segment(3));

        $content = $this->smarty->fetch('cadastros_pacientes/cadastrar_paciente.tpl');
        $this->smarty->assign('content', $content);

        $this->smarty->view('layout');
    }

    // editar cadastro
    public function editar_paciente($id = '') {
        $this->checa_permissao('editar');

        $config = array(
            array(
                'field' => 'nome',
                'label' => 'Nome',
                'rules' => 'required|min_length[3]'
            ),
            array(
                'field' => 'especie',
                'label' => 'Espécie',
                'rules' => 'required'
            ),
            array(
                'field' => 'raca',
                'label' => 'Raça',
                'rules' => 'required'
            ),
            array(
                'field' => 'data_nasc',
                'label' => 'Data de Nascimento',
                'rules' => 'required'
            ),
            array(
                'field' => 'peso',
                'label' => 'Peso',
                'rules' => 'required'
            ),
            array(
                'field' => 'caracteristica',
                'label' => 'Caracteristica',
                'rules' => 'required'
```

```
            ),
            array(
               'field' => 'sexo',
               'label' => 'Sexo',
               'rules' => 'required'
            )
         );

         $this->form_validation->set_rules($config);

         if ($this->form_validation->run() == TRUE) {
            if ($id == '')
               $this->erro();

            $post = $this->input->post();

            $data_invertida = explode('/', $post['data_nasc']);

            $post['data_nasc'] = $data_invertida[2] . '-' . $data_invertida[1] . '-' . $data_invertida[0];

            $this->model_banco->updateTable('pacientes', $post, 'id', $id);

            $this->smarty->assign('sucesso', '<div class="alert alert-success">Cadastro alterado com
sucesso!</div>');
         }

         $consulta_pacientes = $this->model_banco->buscar_tudo("pacientes where id = '$id'");
         $this->smarty->assign('paciente', $consulta_pacientes);

         if (count($consulta_pacientes) == 0)
            $this->erro();

         $consulta_especies = $this->model_banco->buscar_tudo_sort('especie', 'nome', 'asc');
         $this->smarty->assign('especies', $consulta_especies);

         $content = $this->smarty->fetch('cadastros_pacientes/editar_cadastro.tpl');
         $this->smarty->assign('content', $content);

         $this->smarty->view('layout');
      }

      //função para deletar pacientes
      public function deletar_paciente($id) {
         $this->checa_permissao('excluir');

         $consulta_clientes = $this->model_banco->deletar("pacientes", 'id', $id);

         $this->smarty->assign('sucesso', '<div class="alert alert-success">Paciente deletado com
sucesso!</div>');
         $this->lista_clientes();
      }
```

```
    public function ficha_animal($id = '') {
        $this->checa_permissao('visualizar');

        if ($id == '')
            $this->erro();

        // aqui pega os dados do paciente
        $consulta_pacientes = $this->model_banco->sql("select p.*, e.nome as nome_especie, c.nome
as nome_cliente from pacientes p join especie e on p.especie = e.id join clientes c on c.id =
p.idclientes where p.id = '{$id}'");
        $this->smarty->assign('paciente', $consulta_pacientes);

        // aqui carrega as fichas clinicas
        $fichas_clinicas = $this->model_banco->buscar_tudo("ficha_clinica where idpacientes = '{$id}' ");
        $this->smarty->assign('fichas_clinicas', $fichas_clinicas);

        $content = $this->smarty->fetch('cadastros_pacientes/ficha_animal.tpl');
        $this->smarty->assign('content', $content);

        $this->smarty->view('layout');
    }

}
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

class Fichas_clinicas extends Core_Seguro {

    public $_nivel = 'fichas_clinicas';

    public function __construct() {
        parent::__construct();

        $menu_esq = $this->smarty->fetch('blocos/menu_esq.tpl');
        $this->smarty->assign('menu_esq', $menu_esq);
    }

    // função default do controller
    public function index() {
        $content = $this->smarty->fetch('fichas_clinicas/inicio.tpl');
        $this->smarty->assign('content', $content);

        $this->smarty->view('layout');
    }

    // aqui mostra a ficha clinica especifica do paciente
    public function ficha_clinica($id = '') {
        $this->smarty->assign('id_ficha_clinica', $id);

        $ficha_clinica = $this->model_banco->buscar_tudo("view_ficha_clinica where id = '{$id}'");
```

```
        $this->smarty->assign('ficha_clinica', $ficha_clinica);

        echo $this->smarty->fetch('fichas_clinicas/ficha_clinica.tpl');
    }

    // aqui imprime a ficha clinica especifica do paciente
    public function ficha_clinica_impressao($id = '') {
        $ficha_clinica = $this->model_banco->buscar_tudo("view_ficha_clinica where id = '{$id}'");
        $this->smarty->assign('ficha_clinica', $ficha_clinica);

        echo $this->smarty->fetch('fichas_clinicas/ficha_clinica_impressao.tpl');
    }

}

<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

class Inicial extends Core_Seguro {

    public function __construct() {
        parent::__construct();
    }

    public function index() {
        $this->smarty->view('layout');
    }

}

<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

class Login extends CI_Controller {

    public function __construct() {
        parent::__construct();
    }

    public function index() {
        $config = array(
            array(
                'field' => 'usuario',
                'label' => 'Usuário',
                'rules' => 'required|min_length[3]'
            ),
            array(
                'field' => 'senha',
```

```
            'label' => 'Senha',
            'rules' => 'required|min_length[3]'
        )
    );

    $this->form_validation->set_rules($config);

    if ($this->form_validation->run() == TRUE) {
        $valida = $this->model_banco->buscar_tudo("usuarios where login = '{$this->input-
>post('usuario')}' and senha = '".md5($this->input->post('senha'))."'");

        if (count($valida) != 0){
            $this->session->set_userdata('logado', TRUE);
            $this->session->set_userdata('id_logado', $valida[0]->id);
            $this->session->set_userdata('nivel_logado', $valida[0]->nivel);
            $this->session->set_userdata('nome_logado', $valida[0]->nome);
            redirect('inicial');
        } else {
            $this->smarty->assign('retorno', '<div class="alert alert-error">Login ou senha
incorretos</div>');
        }
    }

    $this->smarty->view('login');
  }

  public function sair() {
    $this->session->sess_destroy();
    redirect('login');
  }

}

<!DOCTYPE html>
<!--[if lt IE 7]>      <html class="no-js lt-ie9 lt-ie8 lt-ie7"> <![endif]-->
<!--[if IE 7]>         <html class="no-js lt-ie9 lt-ie8"> <![endif]-->
<!--[if IE 8]>         <html class="no-js lt-ie9"> <![endif]-->
<!--[if gt IE 8]><!--> <html class="no-js"> <!--<![endif]-->
  <head>
    <base href="{base_url()}" />
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Medicina Veterinária</title>
    <link rel="stylesheet" href="assets/padrao/css/style.css" type="text/css" />
    <link rel="stylesheet" href="assets/padrao/css/bootstrap.css">
    <link rel="stylesheet" href="assets/padrao/css/bootstrap-responsive.min.css">
    <script src="assets/padrao/js/jquery-1.9.0.js"></script>
    <script src="assets/padrao/js/bootstrap.min.js"></script>
    <script src="assets/padrao/js/jquery.mask.min.js"></script>
    <script src="assets/padrao/js/jquery.dataTables.js"></script>
    <script src="assets/padrao/js/scripts.js"></script>
  </head>
  <body>
```

```
    <div class="row-fluid" id="topo">
      <div class="span5">
        <img src="assets/padrao/img/logo.png" alt="Logo" title="Logo" id="logo" />
      </div>
      <div class="offset4 span3">
        <p class="hidden-phone">
          <br>
          <img src="assets/padrao/img/avatar.png" alt="Avatar" title="Avatar" class="img-
rounded">
        </p>
        Seja bem vindo <strong><em>{$nome_logado}, <a
href="login/sair">sair</a></em></strong>

      </div>
    </div>
    <div class="navbar">
      <div class="navbar-inner">
        <div class="container">
          <button type="button" class="btn btn-navbar" data-toggle="collapse" data-target=".nav-
collapse">
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
          </button>
          <div class="nav-collapse collapse">
            <ul class="nav">
              <li class="active"><a href="#">Página Inicial</a></li>
              <li class="dropdown">
                <a href="{base_url()}cadastros" class="dropdown-toggle" data-
toggle="dropdown">Cadastros <b class="caret"></b></a>
                <ul class="dropdown-menu">
                  <li><a href="{base_url()}cadastros/lista_clientes">Clientes</a></li>
                  <li><a href="{base_url()}cadastros_pacientes/lista_clientes">Pacientes</a></li>
                  <li><a href="#">Medicamentos</a></li>
                </ul>
              </li>
              <li><a href="#">Agendamento</a></li>
              <li><a href="#">Medicamentos</a></li>
              <li><a href="#">Lembretes</a></li>
              <li class="dropdown">
                <a href="#" class="dropdown-toggle" data-toggle="dropdown">Relatórios <b
class="caret"></b></a>
                <ul class="dropdown-menu">
                  <li><a href="#">Clientes</a></li>
                  <li><a href="#">Pacientes</a></li>
                  <li><a href="#">Medicamentos</a></li>
                  <li><a href="#">Entrada</a></li>
                  <li><a href="#">Saida</a></li>
                </ul>
              </li>
              <li><a href="#">Configurações</a></li>
            </ul>
```

```
            </div><!--/.nav-collapse -->
          </div>
        </div>
      </div>
      {$content|default:'<div class="row-fluid"><div class="offset1 span10"><div class="hero-
unit"><h1>Seja bem vindo!</h1><p>Para navegar pelo sistema utilize o menu
acima</p></div></div></div>'}
      <footer class="footer">Sistema desenvolvido por: <a href="mailto:
felipedonizetemacedo@hotmail.com">Felipe Donizete Macedo</a>.</footer>
  </body>
</html>

<div class="container-fluid">
  <div class="row-fluid">
    <div class="span3">
      {$menu_esq}
    </div><!--/span-->
    <div class="span9">
      <div class="hero-unit">
        <h1>Ops!</h1>
        <p>Você não tem permissões para acessar está página, contate o administrador!</p>
      </div>
    </div>
  </div>
</div>

<div class="container-fluid">
  <div class="row-fluid">
    <div class="span3">
      {$menu_esq}
    </div><!--/span-->
    <div class="span9">
      <form class="bs-docs-example" action="" method="post">
        <fieldset>
          <legend>Editar Paciente
            <a href="cadastros_pacientes/lista_clientes" class="btn pull-right">Voltar</a>
          </legend>
          <div class="row-fluid">
            <div class="span4">
              {$sucesso|default:''}
              {validation_errors('<div class="alert alert-error">', '</div>')}
              <label>Nome: </label>
              <input type="text" name="nome" value="{$paciente[0]->nome|default:''}"
class="span12">
              <label>Sexo: </label>
              <label class="radio">
                <input type="radio" name="sexo" id="optionsRadios1" value="M" {if $paciente[0]-
>sexo == "M"} checked="checked" {/if} >
                Masculino
              </label>
              <label class="radio">
```

```
                    <input type="radio" name="sexo" id="optionsRadios2" value="F" {if $paciente[0]->sexo == "F"} checked="checked" {/if} >
                    Feminino
                  </label>
                  <label>Espécie: </label>
                  <select name="especie" class="span12">
                    {foreach from=$especies item=value}
                      <option value="{$value->id}" {if ($value->id == $paciente[0]->especie)} selected="selected" {/if}>{$value->nome}</option>
                    {/foreach}
                  </select>
                  <label>Raça: </label>
                  <input type="text" name="raca" value="{$paciente[0]->raca|default:''}" class="span12">
                  <label>Data de Nascimento: </label>{$data = explode('-', $paciente[0]->data_nasc|default:'')}
                  <input type="text" name="data_nasc" value="{$data[2]}/{$data[1]}/{$data[0]}" class="data span12">
                  <label>Peso: </label>
                  <input type="text" name="peso" value="{$paciente[0]->peso}" class="span12">
                  <label>Caracteristica: </label>
                  <textarea name="caracteristica" class="span12" rows="5">{$paciente[0]->caracteristica|default:''}</textarea>

                  <a href="cadastros_pacientes/lista_clientes" class="btn btn-danger">Cancelar</a>
                  <button type="submit" class="btn btn-success">Salvar</button>
                </div>
              </div>
            </fieldset>
          </form>
        </div>
      </div>
    </div>
```