Sistema Online para Controle Integrado de Atividades em Consultórios Veterinários e Pet Shops

Robson Pereira Machado, Ana Paula Canal, Alexandre de Oliveira Zamberlan

Sistemas de Informação – Centro Universitário Franciscano
CEP 97.010-032 – Santa Maria – RS – Brasil

rmachczew@gmail.com, apc@unifra.br, alexz@unifra.br

Abstract. *This paper describes the design and development of two systems. First an application for Android that performs scanning QR Codes and manipulate an integrated database, allowing Veterinary Offices and Pet Shops control the activities. Second an online system, built on HTML5, PHP and JavaScript, able to report the progress of services to pet while in baths, shears or vaccines. Both systems are integrated into a database implements for Dorneles (2016), which fits as a case study.*

Resumo. *Este artigo descreve o projeto e o desenvolvimento de dois sistemas informatizados. Primeiro, um aplicativo para Android que realiza leitura de QR Codes e manipula uma base de dados integrada, possibilitando que Consultórios Veterinários e Pet Shops controlem as atividades por meio do smartphone. Segundo, um sistema online, construído em HTML5, PHP e JavaScript, capaz de mostrar o andamento dos serviços prestados aos animais de estimação como banhos, tosas ou vacinas. Ambos os sistemas são integrados a uma base de dados implementados por Dorneles (2016), que serve como estudo de caso.*

1. Introdução

Com o aumento de animais inseridos no âmbito familiar, tornando-se membros da família, surge a insegurança em deixá-los em qualquer estabelecimento ou com qualquer pessoa. Portanto, para melhorar a forma de como os animais são atendidos por Consultórios Veterinários (intervenções cirúrgicas) e *Pet Shops* (serviços de banho e tosa) surge a necessidade de tornar o estabelecimento mais transparente e confiável ao cliente [Toviansky 2014].

Com a quantidade dos recursos tecnológicos disponíveis no mercado, as informações estão mais disponíveis e de rápido acesso, tanto por meio de dispositivos *desktops*, quanto por dispositivos móveis. Assim, as empresas precisam tornar a realização de seus serviços visíveis aos clientes [Rezende e Abreu 2011].

Sendo assim, este trabalho teve como objetivo projetar e desenvolver um sistema capaz de proporcionar visibilidade de serviços prestados por uma empresa aos clientes proprietários de animais de estimação.

O sistema sugerido está inserido no contexto de Sistemas de Informação, principalmente na área de Sistemas Gerencias de Relacionamento com o Cliente. Essa categoria foca na satisfação e fidelização dos clientes [Laudon e Laudon 2014].

Para isso, houve a integração da tecnologia de Internet sem fio (*Wireless*) com uma base de dados. Essa integração possibilitou que o sistema fosse capaz de controlar

e revelar *online* o andamento das atividades em que os animais de estimação estão sendo submetidos. Por conseguinte, o sistema garante que o proprietário do animal acompanhe todos os processos realizados e saiba quando esse estiver pronto sem a necessidade de ser contatado por telefone. Ademais, o sistema possibilita a visualização de diversos relatórios do animal em relação ao atendimento, mesmo quando ele não esteja mais no estabelecimento.

O texto está dividido em cinco seções. A primeira contempla o resumo e a introdução. A segunda seção apresenta o referencial teórico, com conceitos, tecnologias e artigos correlatos que auxiliaram no desenvolvimento do projeto. A terceira seção descreve o projeto, detalhando seu desenvolvimento por meio de uma metodologia de *software*. A quarta seção aborda a conclusão, com os resultados obtidos e os trabalhos futuros, que podem servir de citação para novas propostas. Por fim, é apresentada a quinta seção, que contém as referências bibliográficas utilizadas no desenvolvimento do trabalho.

2. Referencial Teórico

Nesta seção, são abordados conceitos referentes ao contexto de Sistemas de Informação, as tecnologias utilizadas na construção do projeto, bem como trabalhos correlatos que auxiliaram a pesquisa.

2.1. Sistemas de Informação

Segundo Laudon e Laudon (2014), um Sistema de Informação (SI) é um conjunto organizado de componentes que servem para coletar, processar, armazenar e distribuir informações devidamente tratadas, auxiliando na tomada de decisões e no controle da organização pelos profissionais de Tecnologia da Informação (TI).

Sistemas de Informação exercem uma grande influência na administração e vice-versa, havendo uma grande gama de conhecimentos da administração e de negócios no contexto. A partir desse conceito, também se pode dizer que SI não é uma tecnologia, mas o resultado dela [Rezende e Abreu 2011].

Existem diversos tipos de SI, podendo ser classificados a partir do ponto de vista empresarial, de acordo com a sua utilização e o tipo de resultado no processo de tomada de decisões (muitos desses sendo apenas variações de outros). Com isso, pode-se dizer que os tipos mais usuais são: Sistemas de Processamento de Transações (SPTs); Sistemas de Informação Gerenciais (SIG); Sistemas de Apoio à Decisão (SAD); Sistemas de Apoio ao Executivo (SAE); Sistemas Integrados (*Enterprise Resource Planning* – ERP); Sistemas de Gestão da Cadeia de Suprimentos (*Supply Chain Management* – SCM); Sistemas de Gerenciamento do Relacionamento com o Cliente (C*ustomer Relationship Management* – CRM) [Laudon e Laudon 2014].

Assim, a partir dessas definições, acredita-se que o CRM é o que mais se encaixa no projeto a ser descrito, pois busca uma estratégia no negócio e a antecipação às necessidades dos clientes, objetivando a satisfação e a fidelização desses. Todavia, o projeto também se enquadra como SIG, por permitir a visualização de diversos relatórios sobre o animal de estimação e os serviços realizados. Com isso, ambos os tipos de SI ajudam a empresa no contentamento e na retenção de mais clientes, no aumento da receita e um serviço de melhor qualidade.

2.2. Tecnologias Usadas no Projeto

Para o desenvolvimento da plataforma móvel, foi utilizado o sistema operacional Android, que se encarrega de gerenciar a memória, os processos, *threads*, segurança dos arquivos e pastas, redes e *drivers*, cujo *kernel* (responsável pela interação entre o *hardware* e o *software*) é baseado no sistema operacional Linux. Dessa forma, permitiu que o novo sistema herdasse diversas funcionalidades já estruturadas [Lecheta 2011]. Além disso, foi utilizada a biblioteca (ferramentas da linguagem) *Zebra Crossing* (ZXing). Essa biblioteca de codificação e decodificação de códigos de barras tradicionais e QR *Codes* permite transformar uma *string* (vetor de caracteres) em uma matriz de código binário [Costa et al. 2011].

Para a plataforma Web, foram usadas as linguagens de marcação HTML5 (*Hypertext Markup Language*) e CSS3 (*Cascading Style Sheets*). HTML5 especifica como textos, gráficos, vídeos e sons serão organizados em um site. CSS3 define como os elementos contidos no código serão exibidos, possibilitando que colunas no *layout* sejam distribuídas de forma dinâmica [Raible 2013].

Também, houve o uso do *framework jQuery* e da linguagem PHP. Segundo Silva (2013), *jQuery* é uma biblioteca *JavaScript* (linguagem de programação baseada em *scripts*) que facilita e simplifica a escrita de códigos, mesmo àqueles sem experiência. A linguagem PHP é uma linguagem de *scripts* com um conjunto de funções, acesso a bancos de dados relacionais como MySQL, possibilitando a criação desde páginas Web simples até aplicações complexas, com o uso ou não da teoria Orientação a Objetos [Dall'Oglio 2009].

2.3. Trabalhos Relacionados

No trabalho proposto por Lopez (2012), é apresentado o desenvolvimento de um sistema de pedidos, utilizando o Sistema Operacional Android, capaz de acompanhar o registro de serviços em empresas do setor gastronômico. No desenvolvimento do sistema do painel administrativo e da camada de integração foi utilizada a linguagem PHP junto com o banco de dados MySQL. O QR *Code* foi usado na aplicação do cliente, que reproduz os números das mesas do estabelecimento. O cliente pode acessar as informações, pedidos ou serviços de uma determinada mesa, com a ajuda de um leitor QR *Code* instalado no *smartphone*.

O trabalho proposto por Costa, Gomes, Vasconcelos e Wesche (2011), buscou-se melhorar o controle de mercadorias e bens patrimoniais com o auxilio da tecnologia QR *Code*. Para o desenvolvimento do sistema foram utilizados a linguagem *Java* com a tecnologia Web JSF integrada as bibliotecas *PrimeFaces* e ZXing, ambos com acesso ao banco de dados MySQL. O projeto proposto conseguiu atingir o objetivo inicial de proporcionar às empresas e instituições um sistema gestor que facilite o gerenciamento dos bens patrimoniais na entidade.

No último trabalho, proposto por Aires, Orlovski e Ribeiro (2012), o objetivo foi apresentar um *software* para desenvolvimento Web para o gerenciamento e controle de academias. Para o desenvolvimento do sistema foram utilizados a linguagem PHP e os *frameworks Bootstrap* e *jQuery*. Além desses, foram usados HTML5, CSS3 e o banco de dados MySQL. Os resultados obtidos foram o controle de clientes, mensalidades, medidas corporais, professores e agendamentos. O projeto procurou gerenciar o controle

das atividades realizadas nas academias. Entretanto, o sistema não abrange o cliente como parte do negócio.

Por meio dos trabalhos correlatos, a pesquisa e o desenvolvimento do projeto tiveram um fluxo inicial de construção definida, bem como as ferramentas a serem utilizadas na realização do projeto. Em comum com o 1º trabalho correlato, foi usado à linguagem PHP para a integração da plataforma móvel com a plataforma Web. Como o 2º trabalho, foi utilizado a biblioteca ZXing. O 3º trabalho auxiliou no desenvolvimento da plataforma Web com uso de PHP, do *framework jQuery* e da linguagem CSS.

2.4. Metodologia de Desenvolvimento de Software

Uma metodologia é definida como um conjunto de boas práticas. No contexto de *software* a definição é a mesma, ou seja, o programador solitário foi substituído por uma equipe, e essa foi dividida objetivando que cada profissional se concentrasse numa parte especifica do desenvolvimento do projeto [Pressman 2010].

Há duas definições para metodologias de desenvolvimento de *softwares*, que são [Ferreira 2012]:

Metodologias Tradicionais: são orientadas à documentação, ou seja, o *software* precisa ser todo planejado e documentado antes do desenvolvimento, para diminuir os custos gerados por alterações e correções posteriores;

Metodologias Ágeis: utilizam indivíduos e interações ao invés de processos e ferramentas; *software* executável ao invés de documentação; colaboração do cliente ao invés de contratos; respostas rápidas as mudanças ao invés de seguir planos.

Com as definições, a metodologia escolhida para o desenvolvimento do projeto foi a *Feature Driven Development* (FDD), que é uma metodologia ágil utilizada como ferramenta para projetos orientados a objetos, com desenvolvimento iterativo e incremental, promovendo a colaboração entre desenvolvedores e *stakeholders* (atores, interessados, participantes) do planejamento até os testes do produto [Pressman 2010].

A FDD é dividida em cinco processos, separada por duas fases: Concepção e Planejamento, e Construção. A primeira fase possui três processos com objetivo de desenvolver um modelo geral de escopo, listar e planejar as características do sistema. Já na segunda fase, é tratado o desenho e a construção do projeto [Pressman 2010].

3. Descrição do projeto

Inicialmente, foi desenvolvido um aplicativo para plataforma móvel no sistema operacional Android capaz de realizar a leitura de QR *Codes*, verificá-lo em um banco de dados, implementado no trabalho desenvolvido por Dorneles (2016), e alterar o *status* (coluna da tabela atendimentos) da atividade caso o código seja encontrado. Dessa forma, é alcançado o controle das atividades no estabelecimento.

Após, houve o desenvolvimento de um site (plataforma Web) integrado ao mesmo banco de dados, que permite aos clientes (por meio de *login* com senha de acesso disponibilizado durante o cadastro de atendimento) poderem acessar e acompanhar o andamento dos serviços realizados nos animais de estimação em tempo *online*.

Além disso, o sistema cruza dados disponíveis na base de dados para gerar diversos relatórios que são: i) quantidade de atendimentos no ano; ii) média mensal de atendimentos; iii) quantidade de banhos por animal de estimação; iv) quantidade de banhos e tosas; v) quantidade de vacinas; vi) data do último atendimento realizado pelo cliente; vii) data da última vacinação e a descrição desta. Todos os relatórios somente são disponibilizados aos usuários que estejam acessando o sistema, pois o sistema relaciona todas as funcionalidades com o usuário correspondente ao atendimento.

Há também disponibilidade de envio de sugestões à empresa com um *feedback* dos serviços prestados, possibilitando na melhoria contínua do negócio.

Ambos os sistemas (plataforma móvel e plataforma Web) têm acesso autorizado para manipulação da tabela "funcionarios" (coluna cpf e nome) e "atendimentos" (coluna nome, funcionario, senha, status, servico, dataFim).

A cada troca de *status*, o dono do animal é notificado por e-mail do andamento do serviço. Por exemplo, se está na gaiola, no banho, na tosa, na secagem e pronto. Esse último *status* faz com que o sistema envie um e-mail avisando da conclusão do serviço para que o cliente possa buscar seu animal sem a necessidade de ser contatado pelo estabelecimento.

3.1. Análise de Requisitos e Explanação dos Processos FDD

Para uma melhor compreensão, tanto os requisitos quanto os diagramas foram divididos em plataforma móvel e em plataforma Web. Foi estruturado assim, pois os sistemas não se comunicam diretamente sem que haja o auxílio de outro sistema com uma base de dados pronta e configurada para que se possa receber as informações geradas.

Os requisitos funcionais levantados para o projeto são:

Plataforma Móvel**:** RF01 – Solicitar Leitura de Código; RF02 – Alterar Banco de Dados.

Plataforma Web: RF01 – Efetuar *Login*; RF02 – Acompanhar Atividade; RF03 – Visualizar *Status*; RF04 – Consultar Relatório de Atendimento Anual; RF05 – Consultar Relatório de Atendimento Mensal; RF06 – Consultar Relatório de Banhos; RF07 – Consultar Relatório de Tosas; RF08 – Consultar Relatório de Vacinas; RF09 – Cadastrar Sugestão.

Os requisitos não funcionais levantados são:

Plataforma Móvel: RNF01 – É necessária que haja um dispositivo móvel com sistema operacional Android; RNF02 – É necessária conexão com a Internet para acessar o sistema; RNF03 – É necessária que tenha uma câmera com foco automático; RNF04 – A conexão com a base de dados deve ser protegida pela aplicação; RNF05 – O sistema será desenvolvido com a linguagem *Java*, e a biblioteca ZXing.

Plataforma Web: RNF01 – É necessária conexão com a Internet para acessar o sistema; RNF02 – É necessário um navegador atual capaz de interpretar HTML5 e CSS3; RNF03 – O usuário deve digitar senha válida para acessar o site; RNF04 – A conexão com a base de dados deve ser protegida pelo sistema; RNF05 – O sistema será desenvolvido com a linguagem PHP, em conjunto das linguagens HTML5 e CSS3, e dos *frameworks jQuery*.

3.1.1. Desenvolver um Modelo Geral

Segundo Lucca (2002), o desenvolvimento de um modelo geral é um estudo sobre o escopo do sistema e seu conteúdo, que abrange todo o projeto. Assim, para o projeto foi diagramado os fluxos de dados usando a notação BPMN (modelagem de processos de negócios) com o uso do *software* Bizagi Modeler.

Primeiramente, foi desenvolvida a modelagem para a plataforma móvel, apresentado na Figura 1, que mostra o fluxo de execução com a ativação da *Webcam*, a leitura do QR *Code* e a integração com a base de dados. A seguir, foi desenvolvido o modelo geral para a plataforma Web, conforme a Figura 2, que possibilita a visualização do atendimento prestado pelo estabelecimento bem como o acesso aos relatórios e a funcionalidade de sugestão.

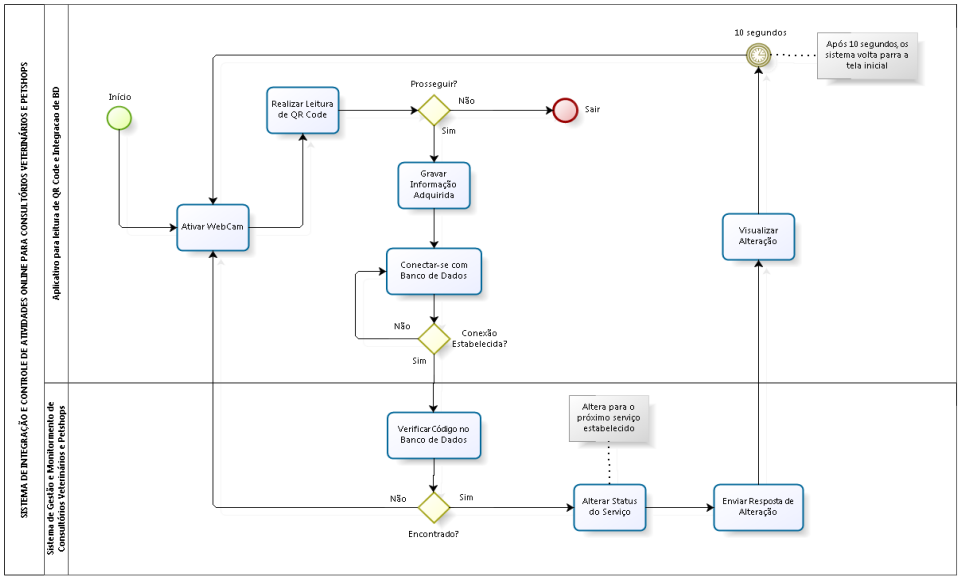

Figura 1: Diagrama de Fluxo de Dados da Plataforma Móvel.

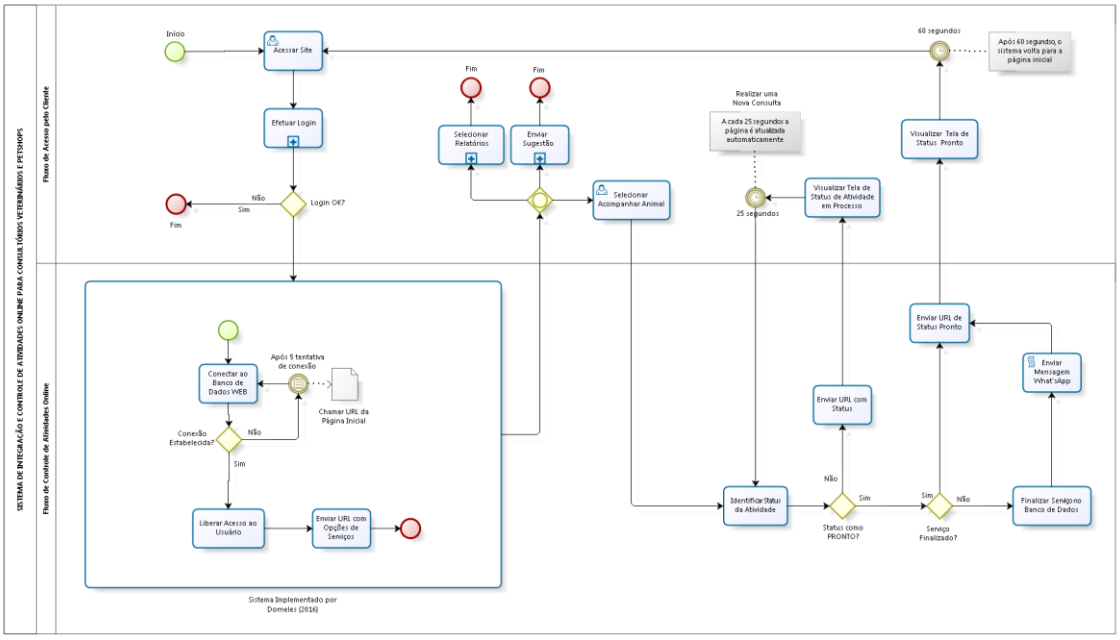

Figura 2: Diagrama de Fluxo de Dados da Plataforma Web.

3.1.2. Construir a lista de funcionalidades

A construção da lista de funcionalidades é a atividade inicial do projeto, permitindo que, por meio do conhecimento adquirido com a análise do modelo geral do sistema, se identifique todas as funcionalidades que satisfaçam aos requisitos analisados [Lucca 2002].

A metodologia FDD não cita a utilização de diagramas de Caso de Uso, todavia, com seu uso as funcionalidades que serão disponibilizadas aos usuários são apresentadas de forma mais simples de serem visualizadas. Com a construção do modelo geral do projeto, foi possível desenvolver o diagrama de Caso de Uso referente à plataforma móvel, ilustrado na Figura 3, que ilustra o acesso ao aplicativo, à leitura do QR *Code* e a alteração da base de dados. O diagrama referente à plataforma Web, Figura 4, mostra todas as opções disponibilizadas ao usuário ao acessar o sistema.

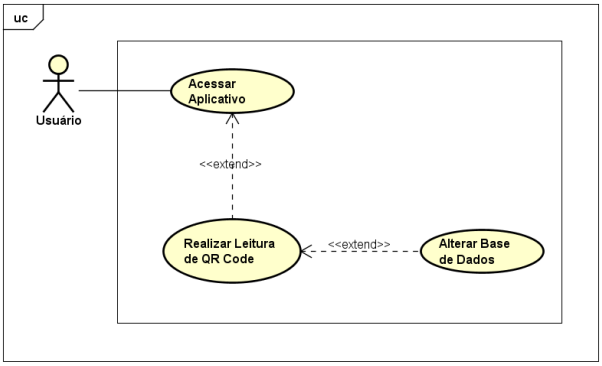

Figura 3: Diagrama de Caso de Uso da Plataforma Móvel.

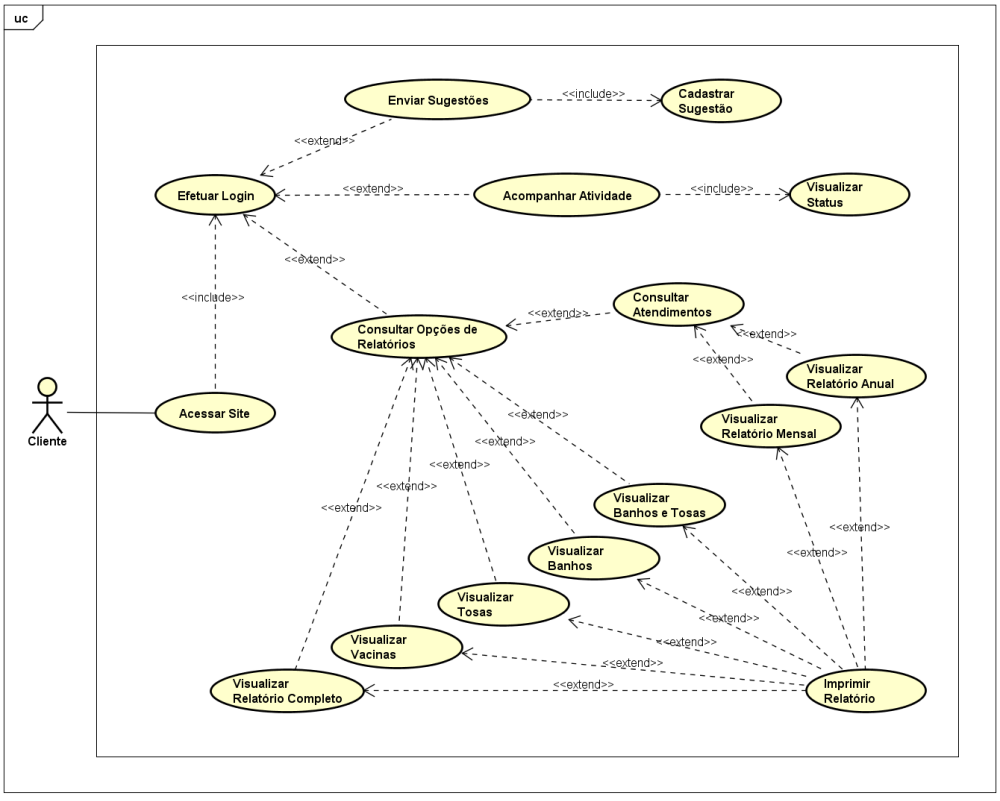

Figura 4: Diagrama de Caso de Uso da Plataforma Web.

A partir dos diagramas gerados anteriormente, foi possível o desenvolvimento dos diagramas de classes para o projeto. Assim, a Figura 5 mostra o diagrama da plataforma móvel, que tem como principal classe a integração do sistema, pois é por meio desta que todo o sistema é controlado. O seu mau funcionamento impede a visualização e controle do atendimento. Em seguida, a Figura 6 abrange o diagrama de classe para a plataforma Web, que tem como classe principal Acompanhar Atividade, mostrando o andamento do animal no estabelecimento ao cliente.

Os diagramas de classes dispostos a seguir são uma simplificação dos originais, utilizados dessa forma para facilitar a compressão da ideia.

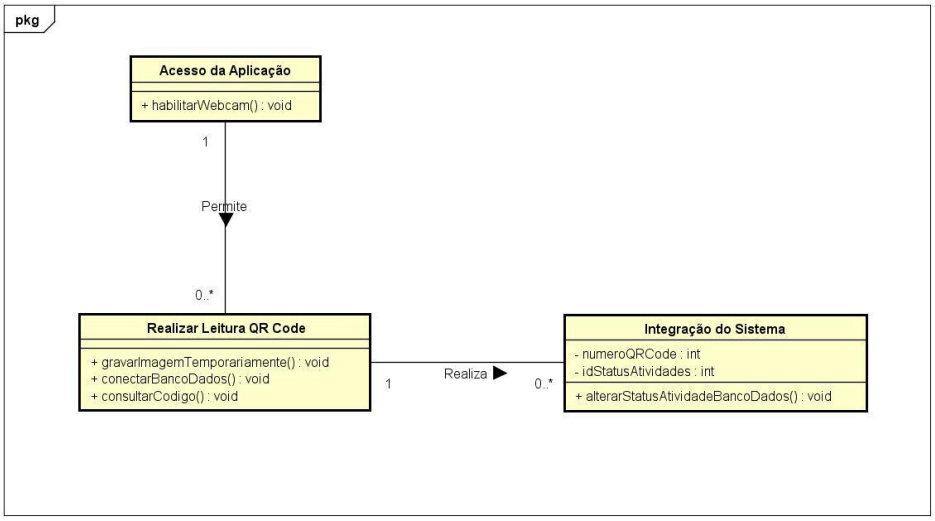

Figura 5: Diagrama de Classe da Plataforma Móvel.

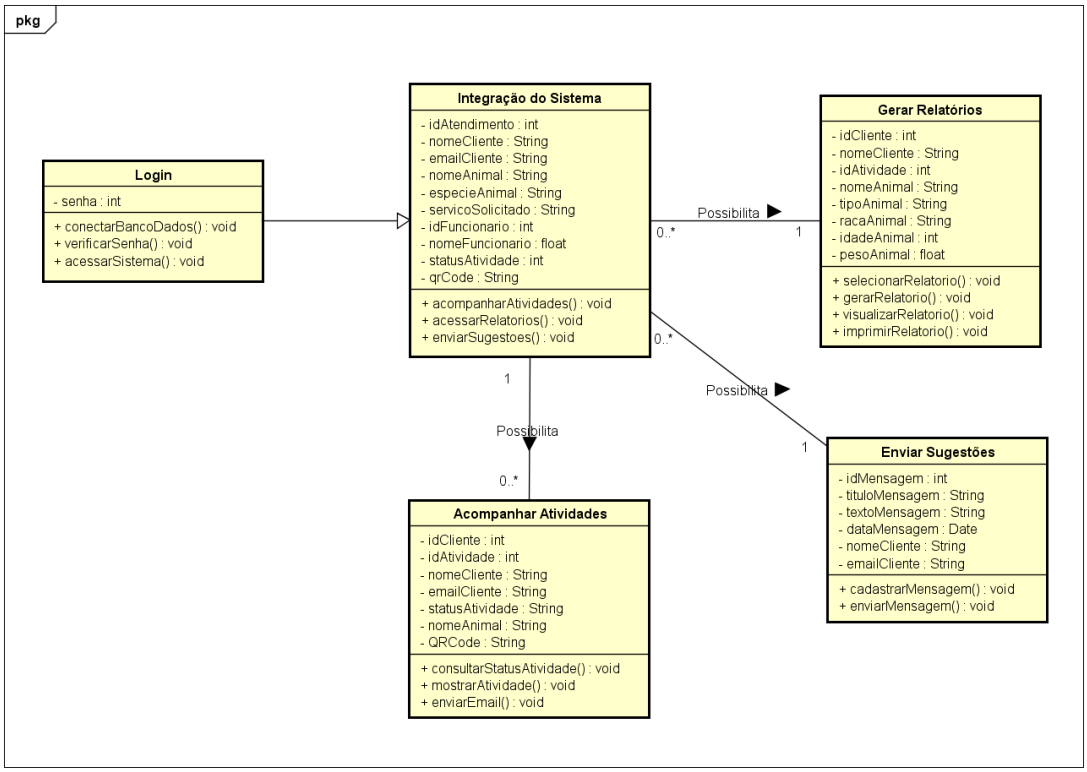

Figura 6: Diagrama de Classe da Plataforma Web.

3.1.3. Planejar por funcionalidade

Nesta seção, é determinada uma ordem de desenvolvimento das funcionalidades, respeitando dependências, a capacidade e a complexidade de cada uma delas [Pressman 2010].

Em função de ser apenas um programador, a atribuição de classes para cada membro da equipe foi desconsiderada. Contudo, as atividades previstas foram listadas: implementar as classes geradas na fase de listagem das funcionalidades; escrever os métodos em suas respectivas classes conforme caso de uso; testar e avaliar; validar.

3.1.4. Projetar por funcionalidade

Segundo Pressman (2010), a partir desta fase, as funcionalidades do sistema são apresentadas aos desenvolvedores para que haja um maior envolvimento com o projeto. Com isso, são projetadas as primeiras telas e especificações da interface, sendo gerados também os diagramas de sequência das funcionalidades, refinando e inspecionando o modelo de objetos originados inicialmente [Lucca 2002]. Conforme a Figura 7 é ilustrada o diagrama referente à leitura de QR *Code*, realizada pela plataforma móvel. A Figura 8 mostra o diagrama para o acompanhamento de animais, na plataforma Web.

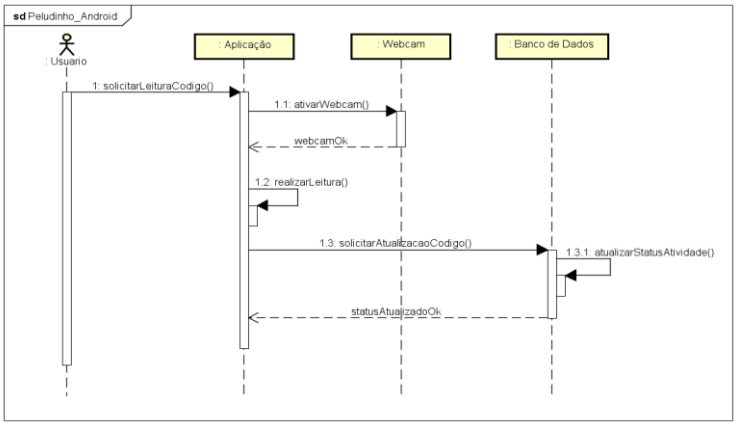

Figura 7: Diagrama de Sequencia da Plataforma Móvel de Leitura de QR *Code*.

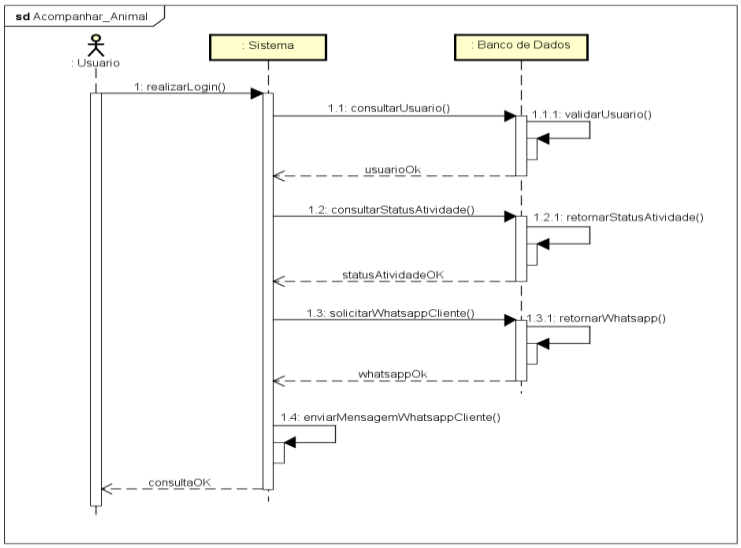

Figura 8: Diagrama de Sequencia para o Acompanhamento de Animais na Web.

Na Figura 9, é possível visualizar a arquitetura de todo o sistema. No lado B da figura, está o desenvolvimento realizado nesse trabalho, sendo inicialmente construída a aplicação móvel, capaz de realizar a leitura de QR *Codes*, atualizar o banco de dados, permitir relatórios de cada funcionário, bem como finalizar o atendimento. Logo, foi construída a interface Web para que o cliente possa acompanhar seu animal, consultar e gerar relatórios, além de enviar sugestões para o aprimoramento do estabelecimento.

No lado A, está o trabalho desenvolvido por Dorneles (2016) o qual foi implementado o sistema de gestão e o banco de dados utilizado por esse projeto.

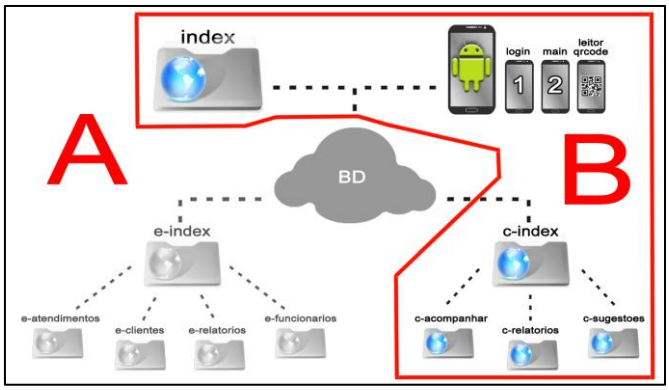

Figura 9: Arquitetura do Sistema.

Para visualizar a implementação do trabalho, a Figura 10 ilustra todas as telas desenvolvidas tanto para a plataforma móvel quanto para a plataforma Web. À esquerda da linha está o sistema Web, onde às telas dispostas da esquerda para a direita mostram a tela de Sugestões, Consultas/Relatórios, Principal (*index*) e Acompanhar. À direita da linha, está a tela de *login* e a tela principal da aplicação Móvel.

Figura 10: À esquerda, Plataforma Web. À direita, Plataforma Móvel.

3.1.5. Construir por funcionalidade

Nesta seção, as funcionalidades partem para a implementação, em que as classes e métodos devem ser implementados, satisfazendo os requisitos para cada funcionalidade. Com isso, o código deve ser inspecionado por meio de testes unitários e se houver uma inspeção bem sucedida, deve seguir para a compilação (*build*). Para concluir o uso da metodologia FDD, após cada funcionalidade ser construída, os processos de Projetar e de Construir devem ser repetidos para cada nova funcionalidade até que todas elas passem por esses processos [Lucca 2002].

A seguir, estão dispostos à implementação (construção) da plataforma móvel, da plataforma Web e concluindo com a avaliação do sistema.

3.1.5.1 Implementação da Plataforma Móvel

Os dois métodos mostrados (Figura 11) são usados para a integração do Android com a biblioteca ZXing, utilizando-se de um botão que chama a classe de captura e assim que a imagem for lida, envia a resposta de volta por meio do objeto *REQUEST_CODE*. Isso é mostrado em um *TextView* chamado de "resultado".

Na Figura 12, é visto o método "button_alterar_pronto" que é usado para alterar a coluna status da tabela de atendimentos. O método envia via *postHttp* o resultado da leitura do QR *Code* realizado pela integração do ZXing junto a palavra "Pronto", para identificar que o serviço está no estado de pronto. Semelhante a esse método, foram criados um para cada estado do *status* a ser passado. Nesse método é usado *Thread* para melhorar o desempenho da aplicação.

```
//Button zxing
public void button_zxing(View v){
    //chamar de volta a tela
    Intent it = new Intent(MainActivity.this, com.google.zxing.client.android.CaptureActivity.class);
    startActivityForResult(it, REQUEST_CODE);
}

@Override
public void onActivityResult(int requestCode, int resultCode, Intent data){
    //acessa o CaptureActivity, metodo handleDecode
    if(REQUEST_CODE == requestCode && RESULT_OK == resultCode){
        resultado.setText(data.getStringExtra("SCAN_RESULT"));
    }
}
```

Figura 11: Integração com a Biblioteca ZXing.

```
public void button_alterar_pronto (View v){
    new Thread(){
        public void run(){
            String pronto = "Pronto";
            postHttp(resultado.getText().toString(), pronto);
        }
    }.start();
}
```

Figura 12: Método que Altera o *Status* do Atendimento.

Na Figura 13, é implementado o método que possibilita a integração do Android com o servidor Web, por meio da linguagem PHP. O método instancia a classe *HttpClient* e com o objeto *HttpPost* há a conexão com a *URL* do servidor direcionado ao arquivo .php, que por meio de uma lista captura os dados recebidos pelo método, "button_alterar_pronto", e os envia ao arquivo .php. Esse possibilita trabalhar com os dados vindos do Android na Web. Assim que o PHP envia resposta, uma *Thread* é executada mostrando a resposta em um *Toast* (provém *feedback* rápido em forma suspensa na tela).

A Figura 14 se assemelha a anterior, sendo usado para o *login* do funcionário. Diferencia-se pelo uso da classe *Bundle*, usado para a passagem de valor da *Activity* de *Login* para a *Activity* Principal, identificando o funcionário e registrando-o como autor do atendimento.

Na Figura 15, é implementado o arquivo em PHP, o qual recebe os dados vindos do Android, realizando assim uma consulta ao banco de dados com o valor contido na variável $qr. Se existir, é atualizada a coluna *status* com o valor recebido por meio da variável $status. Logo, é retornada uma mensagem com resposta a aplicação Android.

Quando o valor passado for “Pronto”, será realizada a atualização do *status*, cadastrando a data e hora final do atendimento e excluindo o QR *Code* do banco para que não possa mais ser modificado.

```
public void postHttp(String qrcode, String status, String funcionario){
    HttpClient httpClient = new DefaultHttpClient();
    HttpPost httpPost = new HttpPost("http://erisonfotos.com.br/c2rsistemas/acesso_bd.php");

    try{
        ArrayList<NameValuePair> valores = new ArrayList<NameValuePair>(3);
        valores.add(new BasicNameValuePair("qrcode", qrcode));
        valores.add(new BasicNameValuePair("status", status));
        valores.add(new BasicNameValuePair("funcionario", funcionario));

        httpPost.setEntity(new UrlEncodedFormEntity(valores));
        final HttpResponse resposta = httpClient.execute(httpPost);

        runOnUiThread(new Runnable(){
            public void run(){
                try {
                    //pega resposta do arquivo server.php
                    Toast.makeText(getBaseContext(), EntityUtils.toString(resposta.getEntity()), Toast.LENGTH_SHORT).show()
                } catch (ParseException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
        });
    }
    catch(ClientProtocolException e){}
    catch(IOException e){}
}
```

Figura 13: Integração com o Servidor Web por meio da Linguagem PHP.

```
try {
    String res = EntityUtils.toString(resposta.getEntity());
    if (res.equals("Acesso Permitido")){
        Toast.makeText(LoginActivity.this, res, Toast.LENGTH_LONG).show();
        itTelaAcessar = new Intent (LoginActivity.this, MainActivity.class);

        //passar valor digitado no login para acitivty principal
        Bundle b = new Bundle();
        b.putString("funcionario", editTextLogin.getText().toString());
        itTelaAcessar.putExtras(b);

        LoginActivity.this.startActivity(itTelaAcessar);
        LoginActivity.this.finish();
    } else {
        Toast.makeText(LoginActivity.this, res, Toast.LENGTH_LONG).show();
    }
```

Figura 14: *Login* de Acesso e Passagem de Valor da *Activity* para outra.

```
<?php
    include "c2rsistemas.com.br/conexao.php";
    $qr = $_POST['qrcode'];
    $status = $_POST['status'];
    $cpfFuncionario = $_POST['funcionario'];
    date_default_timezone_set('America/Sao_Paulo'); //pega data do cadastro e coloca no banco
    $dataFim = date('Y-m-d H:i');

    $sql = mysql_query("SELECT qrcode FROM atendimentos WHERE qrcode = '$qr'") or die(mysql_error());
    $row = mysql_fetch_row($sql);
    $sqlFunci = mysql_query("SELECT nome FROM funcionarios WHERE cpf='$cpfFuncionario'") or die(mysql_error());
    $rowFunci = mysql_fetch_row($sqlFunci);
    $funcionario = $rowFunci[0];
    $qrResult = $row[0];

    if($row > 0){
        if($status == 'Pronto'){
            $sql1 = mysql_query("UPDATE atendimentos SET qrcode='PRONTO', status='$status', dataFim='$dataFim'
            WHERE qrcode='$qrResult'")or die(mysql_error());echo 'Atendimento PRONTO';
        } else {
            $sql2 = mysql_query("UPDATE atendimentos SET funcionario='$funcionario', status='$status', dataFim='$dataFim'
            WHERE qrcode='$qrResult'")or die(mysql_error()); echo 'Status atualizado com sucesso';
        }
    } else { echo 'QR Code nao encontrado.'; }
?>
```

Figura 15: Arquivo PHP para Comunicação entre Android e o Servidor Web.

3.1.5.2. Implementação da Plataforma Web

A partir da Figura 16, está o desenvolvimento do sistema de *login* usado para permitir que o cliente tenha acesso ao Sistema Web. Dessa forma, possibilitando acompanhar o animal, consultar e enviar sugestões sem a necessidade de inserção de seus dados, além de assegurar que ao sair do sistema, seja necessário a inserção da senha de acesso novamente.

Com essa forma de *login*, o usuário não terá um cadastro no sistema. Utilizará a senha gerada no cadastro do atendimento, e a cada novo atendimento uma nova senha será gerada. A senha antiga é apagada após um dia da conclusão do atendimento.

```php
<?php
    if (!empty($_POST) AND (empty($_POST['usuario']) OR empty($_POST['senha']))) {
        header("Location: index.php"); exit;
    }
    mysql_connect('localhost', 'root', '') or trigger_error(mysql_error());
    mysql_select_db('c2rsistemas') or trigger_error(mysql_error());

    $usuario = mysql_real_escape_string($_POST['usuario']);
    $senha = mysql_real_escape_string($_POST['senha']);

    $sql = "SELECT `id`, `usuario` FROM `acesso` WHERE (`usuario` = '". $usuario ."')
            AND (`senha` = '". $senha ."') LIMIT 1";
    $query = mysql_query($sql);
    if (mysql_num_rows($query) == 1) {
        $resultado = mysql_fetch_assoc($query);
        if (!isset($_SESSION)) session_start();
        $_SESSION['UsuarioID'] = $resultado['id'];
        $_SESSION['UsuarioNome'] = $resultado['usuario'];
        header("Location: e-index.php"); exit;
    } else {
        echo '<script type="text/javascript">alert("Usuario inexistente ou senha incorreta.")</script>';
        exit ('<meta http-equiv="refresh" content="1; url=index.php" />');
    }
?>
```

Figura 16: Validação de Acesso ao Sistema Web.

Na Figura 17, é ilustrado a implementação usada em todas as páginas do sistema, garantindo a obrigatoriedade de *login* de acesso.

```php
<?php
if (!isset($_SESSION)) session_start();
if (!isset($_SESSION['UsuarioID'])) {
    session_destroy();
    header("Location: index.php"); exit;
}
?>
```

Figura 17: Condição de Acesso e/ou Destruição da Sessão Criada.

A Figura 18 apresenta a implementação lógica da página para acompanhar o animal, que se assemelham as demais paginas, relatórios e sugestões. Há a inicialização das variáveis e a consulta do atendimento por meio do nome do cliente contido no acesso à página pela sessão inicializada.

Por fim, a Figura 19 apresenta a implementação da condição da espécie do animal cadastrado, mostrando seu estado no andamento do serviço conforme a denominação inserida na coluna *status* da tabela atendimentos. Para simplificar, foi disposta apenas uma condição para cada espécie de animal, havendo outros conforme a denominação dada ao estado do atendimento.

```php
$nome = $_SESSION['UsuarioNome'];

$array[0] = '';
$array[1] = '';
$array[2] = '';

$progresso = '';
$mensagem = 'Ops! Acho que seu animal não está na Pet Shop.';

$sql = "SELECT animal, status, especie FROM atendimentos WHERE nome='$nome'";
$query = mysql_query($sql);
while($row = mysql_fetch_assoc($query)) {
    $array[0] = $row['animal'];
    $array[1] = $row['especie'];
    $array[2] = $row['status'];
}
```

Figura 18: Inicialização das Variáveis e Consulta à Tabela Atendimentos.

```php
if($array[1] == 'Cachorro'){

    if($array[2] == 'Gaiola'){
        $progresso = '<img src="images/cachorro1.png"/>';
        $mensagem = $array[0].' ainda está aguardando, mas logo será atendido(a).';
    }
    ...
}
else if($array[1] == 'Gato'){

    if($array[2] == 'Gaiola'){
        $progresso = '<img src="images/gato1.png"/>';
        $mensagem = $array[0].' ainda está aguardando, mas logo será atendido(a).';
    }
    ...
}
```

Figura 19: Condição de Atualização do *Status* do Atendimento em Andamento.

3.1.5.3. Avaliação do Sistema

Para a obtenção de resultados avaliativos do sistema, aplicou-se um teste de usabilidade com o uso do sistema por algumas pessoas ligadas e não ligadas à área de TI. Os testes seguiram o fluxo: 1) capacitação básica de uso; 2) realização do fluxo principal fornecido pelo sistema; 3) questionário qualitativo para identificar: 3.1) acesso as funcionalidades; 3.2) relatórios gerados; 3.3) disposição dos botões; 3.4) cores; 3.5) desempenho; 3.6) sugestões; 3.7) avaliação do sistema (1 - muito ruins; 2 - ruins; 3 - satisfatórias; 4 - boas; 5 - excelentes).

A partir dos testes realizados, apenas na plataforma Web, identificaram-se questões como: termos usados nos botões não passavam o objetivo das funcionalidades; algumas imagens de fundo estavam diminuindo a performance do sistema; falta de botão fechar no topo das janelas das funcionalidades; a responsividade do sistema não está eficaz; falta de conteúdo para haver mais interatividade com o cliente.

Contudo, para uma avaliação mais cuidadosa, sugere-se um processo que envolva a análise de funcionalidades do sistema por funcionários e clientes de Consultórios Veterinários e *Pet Shops*.

4. Conclusão

O trabalho teve como foco o Sistema de Gerenciamento do Relacionamento com o Cliente (CRM), buscando revisar contextos básicos, processos e ferramentas de construção de sistemas com o objetivo de projetar e implementar um Sistema *Online*

para Controle Integrado de Atividades para Consultórios Veterinários e *Pet Shops*, em que a modelagem e alguns detalhes da implementação foram exibidos.

O sistema desenvolvido denota uma aplicação móvel integrado com um servidor de banco de dados Web atrelado a um site, permitindo o controle das atividades com a leitura de QR *Codes*, a visualização dos procedimentos realizados pelos clientes e do monitoramento desses pelo estabelecimento, ambos por meio da atualização do *status* do serviço solicitado.

Por conseguinte, com a atividade controlada o processo de atendimento da empresa torna-se transparente aos seus clientes, o que possibilita ao dono do animal conhecimento sobre qual processo o seu animal se encontra e quando estará pronto para a retirada, sem a necessidade de contato telefônico. Ainda são disponibilizados diversos relatórios de atendimentos como a quantidade de banhos realizados e a data da última vacinação, e a possibilidade de envio de sugestões ao estabelecimento, o que permite uma maior interação entre a empresa e o consumidor.

Registra-se também, que o estudo teve integração com o trabalho realizado por Dorneles (2016), pois o seu sistema desenvolvido serviu de base de dados ao sistema projetado. Ademais, todas as tecnologias apontadas no projeto do sistema foram utilizadas e combinadas adequadamente.

Para uma prospecção futura de pesquisa, sugere-se a operacionalização de diversas novas funcionalidades e/ou alterações no sistema que podem ser realizadas ou que não foram implementadas, como por exemplo: i) comunicação do sistema com o cliente por meio de SMS (mensagens de textos) e *Whatsapp* (aplicativo de bate-papo); ii) uso de *RFID* (identificação por rádio frequência) ao invés de QR *Codes*, realizando a atualização do *status* automaticamente a cada procedimento realizado; iii) utilização de Notificação *Push*, sistema capaz de enviar notificações de um servidor Web para um usuário cadastrado em dispositivos móveis [Voltolini 2013]; iv) desenvolvimento de aplicativo móvel capaz de possibilitar agendamento de solicitação de atendimento ou busca do animal; v) implementação do aplicativo de leitura de QR *Codes* para IOS da *Apple*; vi) criação de funcionalidade que permita a visualização do histórico de atendimentos ao cliente; vii) disponibilizar uma paleta de cores ou *upload* de imagens ao cliente para que possa tornar a plataforma Web mais interativa.

Referências

Aires, João Antonio, Orlovski, Regiane e Ribeiro, Sergio. (2012) "Desenvolvimento de Sistema de Gerenciamento e Controle para Academias", Curso de Tecnologia em Análise e Desenvolvimento de Sistemas, Faculdade Guairacá, Guarapuava – PR – Brasil.

Costa, João Bruno S., Gomes, Larissa L., Vasconcellos, Fernando Vitor S. e Wesche, Edlyn Faro. (2011) "Sistema de Gestão de Bens Patrimoniais Utilizando QR Code", Instituto de Estudos Superiores da Amazônia (IESAM), Belém – PA – Brasil.

Dall'Oglio, Pablo. (2009), PHP Programando com Orientação a Objetos, 2ª Edição, Editora Novatec.

Dorneles, Caroline R. (2016) "Sistema de Gestão e Monitoramento para Consultórios Veterinários e Pet Shops", Trabalho Final de Graduação do Curso de Sistemas de Informação, Centro Universitário Franciscano – Santa Maria – RS – Brasil.

Ferreira, Joel. (2012) “Metodologias Ágeis de Desenvolvimento de Software e Técnicas de Elicitação de Requisitos Funcionais”, Departamento de Engenharia Informática, Faculdade de Engenharia da Universidade do Porto – Porto – Portugal.

Laudon, Jane P. e Laudon, Kenneth C. (2014), Sistemas de Informação Gerenciais, 11ª Edição, Editora Pearson.

Lecheta, Ricardo R. (2011), Google Android: Aprenda a Criar Aplicações para Dispositivos Móveis com o Android SDK, 2ª Edição, Editora Novatec.

Lopez, Samuel Elias Bravo. (2012) “Gestão de Pedidos em Plataforma Android: Um Sistema para Estabelecimentos do Setor Gastronômico”, Trabalho Final de Graduação do Curso de Sistemas de Informação, Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau – Blumenau – SC – Brasil.

Lucca, Jeff De. (2002) “Feature Driven Development Processes”, http://www.featuredrivendevelopment.com/files/fddprocessesA4.pdf. Outubro.

Pressman, Roger S. (2010), Software Engineering a Practitioner’s Approach, Seventh Edition, McGraw-Hill Education.

Raible, Matt. (2013) “Refreshing Your UI with HTML5, Bootstrap and CSS3”. HTML5 Denver Users Group, Raible Designs Enterprise Open Source Consulting.

Rezende, Denis A. e Abreu, Aline França De. (2011), Tecnologia da Informação Aplicada a Sistemas de Informação Empresariais, 8ª Edição, Editora Atlas.

Silva, Maurício Samy. (2013), jQuery, A Biblioteca do Programador JavaScript, Editora Novatec.

Toviansky, Daniela. (2014) “Bicho de Estimação Virou Membro da Família”. http://exame.abril.com.br/revista-exame-pme/edicoes/69/noticias/o-novo-membro-da-familia. Novembro.

Voltolini, Ramon. (2013) “Por Que as Notificações de Aplicativos são Importantes?”. http://www.tecmundo.com.br/apps/43525-por-que-as-notificacoes-de-aplicativos-sao-importantes-.htm. Abril.