UNIVERSIDADE FEDERAL DO PIAUÍ
CAMPUS SENADOR HELVÍDIO NUNES DE BARROS
CURSO BACHARELADO EM SISTEMAS DE INFORMAÇÃO

FERRAMENTA COMPUTACIONAL DE AUXÍLIO AO SERVIÇO ODONTOLÓGICO – SEOD - UFPI

MICAEL DE ARAUJO BASTOS COSTA

PICOS - PI
2016

MICAEL DE ARAUJO BASTOS COSTA

FERRAMENTA COMPUTACIONAL DE AUXÍLIO AO SERVIÇO ODONTOLÓGICO – SEOD - UFPI

Monografia submetida ao Curso de Bacharelado de Sistemas de Informação como requisito parcial para obtenção de grau de Bacharel em Sistemas de Informação.

Orientadora: Profª. Ma. Patricia Medyna Lauritzen de Lucena Drumond

PICOS - PI
2016

FICHA CATALOGRÁFICA
Serviço de Processamento Técnico da Universidade Federal do Piauí
Biblioteca José Albano de Macêdo

C837f Costa, Micael de Araújo Bastos.

Ferramenta computacional de auxílio ao Serviço Odontológico - SEOD - UFPI / Micael de Araújo Bastos Costa.– 2016.

CD-ROM : il.; 4 ¾ pol. (43 f.)

Monografia (Curso Bacharelado em Sistemas de Informação) – Universidade Federal do Piauí, Picos, 2016.

Orientador(A): Prof. Patrícia Medyna Lauritzen de Lucena Drumond.

1. Sistema Odontológico. 2. Odontograma Digital. 3. Desenvolvimento *Web*. I. Título.

CDD 005.1

FERRAMENTA COMPUTACIONAL DE AUXÍLIO AO SERVIÇO ODONTOLÓGICO – SEOD - UFPI

MICAEL DE ARAUJO BASTOS COSTA

Monografia aprovada como exigência parcial para obtenção do grau de Bacharel em Sistemas de Informação.

Data de Aprovação

Picos – PI, 23 de fevereiro de 2016

Profª. Ma. Patrícia Medyna Lauritzen de Lucena Drumond
Orientadora

Esp. Nonato Rodrigues de Sales Carvalho
Membro

Prof. Esp. Pablo de Abreu Vieira
Membro

Dedico este trabalho a minha família, em especial aos meus pais Carminho e Niceleide e a minhas irmãs Mayara e Angela Monique, por sempre acreditarem no meu potencial e me apoiarem durante todo o tempo.

AGRADECIMENTOS

Agradeço primeiramente a Deus por ter me dado saúde e força para superar as dificuldades para chegar até onde cheguei. A minha família pelo incentivo, carinho e confiança, em especial aos meus pais, pelos ensinamentos, conselhos e pelo esforço em sempre proporcionar aos seus filhos tudo de melhor. A minhas irmãs Mayara e Angela que sempre se fizeram presentes na minha vida, me dando todo o carinho e apoio necessário.

A minha orientadora Patrícia Medyna, pela ajuda constante no amadurecimento do projeto, pelo qual não mediu esforços em compartilhar seu conhecimento e sua experiência que foram cruciais no desenvolvimento do projeto e também por toda a dedicação com que coordena o curso de Sistemas de Informação.

A todos os professores do curso, que foram tão importantes na minha vida acadêmica e no desenvolvimento desta monografia. Aos amigos e colegas, pelo incentivo e pelo apoio constante.

Agradeço também aos meus amigos que fazem parte do Centro de Tecnologia da Informação (CTI), em especial ao Nonato e Jonnison, por sempre estarem disponíveis a me ajudar quando foi preciso. Obrigado a todas aquelas pessoas que sempre torceram por mim, independentemente de estarem longe ou perto, obrigado por cada palavra de incentivo, cada abraço amigo e cada momento partilhado.

“Para se ter sucesso, é necessário amar de verdade o que se faz. Caso contrário, levando em conta apenas o lado racional, você simplesmente desiste. É o que acontece com a maioria das pessoas.

(Steve Jobs)

RESUMO

A utilização de sistemas de informações está aumentando cada vez mais e esse aumento ocorre pela necessidade em um maior controle das informações que são produzidas e mantidas pelas pessoas e organizações. Com isso surgiu a necessidade de um *software* que automatiza as tarefas de cadastro de paciente e a criação de odontograma. Com isso foi proposta a criação de uma ferramenta computacional, ou seja, um sistema para o Serviço Odontológico – SEOD – UFPI, para auxiliar manipulação dos dados dos pacientes, fazendo com que o odontólogo tenha uma ferramenta capaz de fazer o cadastro dos pacientes, e em seguida o Odontograma da arcada dentária do mesmo. O sistema se propõe a deixar um histórico de utilização e procedimentos realizados para cada paciente. Para o desenvolvimento deste trabalho foram utilizadas: a metodologia de desenvolvimento *Scrum*, a modelagem UML, a linguagem de programação Python, o *Framework* de desenvolvimento Django e o *Framework* de *front-end Bootstrap.*

Palavras-chave: Sistema Odontológico, Odontograma Digital, Desenvolvimento Web, Django, Python, JavaScript.

ABSTRACT

The use of information systems is increasing more and this increase is the need for greater control of the information that is produced and maintained by people and organizations. Thus the need has arisen for an automation task was then proposed to create a computational tool, or a system for dental care - SEOD - UFPI to assist manipulation of patient data, so that the dentist has a tool capable of placing the patient, and then the dental chart of the dental arch of the same. The system is proposed to leave a record of usage and procedures performed for each patient. For the development of this work were used: the Scrum development methodology, UML modeling, programming language Python, Django development framework and the Framework Bootstrap front-end.

Keywords: Dental System, Digital dental chart, Web Development, Django, Python, JavaScript.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

CMS	*Content Management System*
CSHNB	Campus Senador Helvídio Nunes de Barros
CSS	*Cascading Style Sheets*
DRY	*Don't Repeat Yourself*
GPL	*General Public License*
HTML	*HiperText Markup Language*
IBGE	Instituto Brasileiro de Geografia e Estatística
IHC	Interação humano-computador
MVC	*Model-View-Controller*
ORM	Object-relational mapping
SAD	Sistema de Apoio a Decisão
SAE	Sistema de Apoio ao Executivo
SEOD	Serviço Odontológico
SIE	Sistemas de Informação Executivas
SIG	Sistema de Informação Gerencial
SQL	*Structured Query Language*
TI	Tecnologia da Informação
UFPI	Universidade Federal do Piauí
UML	*Unified Modeling Language*
XML	Extensible Markup Language
XP	*Extreme Programming*

SUMÁRIO

1 INTRODUÇÃO

Atualmente, as Tecnologias da informação (TI) estão inseridas em diversas áreas e são responsáveis por tornar mais eficiente a maneira de trabalho de uma empresa ou organização. A *Internet* é utilizada em larga escala, sendo responsável por aproximar e melhorar a comunicação e a troca de informações entre as pessoas, independente da hora ou do lugar em que se encontram (CRUZ, 2009).

O uso da Tecnologia da Informação juntamente com os Sistemas de Informação em organizações vem crescendo a cada dia. Lastres e Ferraz (1999), afirmam que a TI tornou-se indispensável tanto na administração pública como também para o setor privado, podendo potencializar o desenvolvimento através dos processos de gestão da organização, e muitas vezes tornando cada vez mais eficientes as tomadas de decisões.

Nas diversas áreas do conhecimento a utilização de sistemas de informação se faz necessária para organização de dados que podem auxiliar na tomada de decisão. No contexto do Serviço Odontológico, um sistema para auxílio na manipulação dos dados poderá agilizar as tarefas, já que há um grande volume de pacientes e estes geralmente retornam ao consultório duas ou três vezes para concluir o tratamento. Com a ausência de um *software* auxiliador não se tem um histórico de atendimento a esse paciente, ficando cada consulta isolada uma da outra, sem contar que os procedimentos realizados no paciente são perdidos devido a essa falta de um sistema de informação.

De acordo com as necessidades identificadas, este trabalho propôs o desenvolvimento de uma ferramenta computacional para auxiliar a manipulação dos dados do serviço odontológico da Universidade Federal do Piauí (UFPI), Campus Senador Helvídio Nunes de Barros. O sistema é uma ferramenta capaz de fazer o cadastro do paciente e o odontograma da arcada dentária do mesmo, processo este que atualmente é feito de forma manual pelo odontólogo. Além do mais, o sistema se propõe a deixar um histórico de utilização e procedimentos realizados para cada paciente.

Para o desenvolvimento deste trabalho foi usada uma linguagem de programação de alto nível, chamada Python, por ser uma linguagem de sintaxe simples e clara, a mesma é uma linguagem de grande crescimento nos últimos

anos, sendo utilizada por grades empresas de Tecnologia. Em conjunto com o Python foi utilizado o Framework de desenvolvimento Django, que conta com uma vasta biblioteca de funções pré-implementadas, e que auxilia na escrita dos códigos.

1.1 OBJETIVO

O objetivo desse trabalho foi desenvolver uma aplicação web, para uso no serviço de odontologia da Universidade Federal do Piauí, Campus Senador Helvídio Nunes de Barros, cujo propósito dessa aplicação é auxiliar os Odontólogos no cadastro das informações pertinentes aos pacientes, bem como as informações a respeito da sua arcada dentaria em forma de odontograma.

1.2 ORGANIZAÇÃO DO DOCUMENTO

O trabalho foi organizado em 4 capítulos:

- **Capitulo 2:** Corresponde ao embasamento teórico necessário para o desenvolvimento do projeto, nesse capitulo relata-se as tecnologias utilizadas durante o desenvolvimento do projeto, descrevendo os principais conceitos e alguns exemplos simples.
- **Capítulo 3:** encontram-se as especificações do sistema de Odontologia desenvolvido, apresentando todos os seus requisitos e diagramas modelados e o detalhamento do seu funcionamento.
- **Capítulo 4**: apresenta a conclusão do sistema e sugestões de trabalhos futuros.

2 REFERENCIAL TEÓRICO

Neste capitulo serão abordados alguns conceitos que foram utilizados para desenvolvimento deste projeto.

2.1 SISTEMAS DE INFORMAÇÃO

Sistemas de Informação é uma combinação estruturada de tecnologias da informação e práticas de trabalho, de forma a permitir o melhor atendimento dos objetivos da organização, por exemplo, sistemas de informação de clientes, sistemas de informação acadêmicas, sistema de informação de estoque, etc.

Segundo Mulbert e Ayres (2005), Sistemas de Informação é um conjunto de componentes, que tem como objetivo, coletar, processar, armazenar e distribuir informações, com o intuito de facilitar a coordenação, o controle, a análise e a visualização dentro de uma organização.

Atualmente existe uma variedade de sistemas de informação, devido a praticidade em manipular os dados e gerar informações com mais agilidade para tomada de decisão. Além disso, não existe um grande sistema que atenda a todas as necessidades que surgem, e sim, a existência de diferentes tipos de sistemas para atender diferentes níveis de problemas.

Segundo Mulbert e Ayres (2005), os sistemas podem ser classificados de vários modos, não existindo uma única classificação rígida. É comum essa classificação ser dada por meio de características da organização. As organizações são divididas em níveis de hierarquia, onde cada nível possui um grupo de profissionais com responsabilidades e funções específicas.

Conforme Gordon e Gordon (2006), existem diversos tipos de Sistemas, que podem estar inseridos em diversas áreas. Quando se trata de sistemas que são aplicados em instituições ou no mercado de trabalho, os mais conhecidos são: SAD (Sistemas de Apoio à Decisão); SIE (Sistemas de Informação Executivas); SIG (Sistemas de Informação Gerencial); entre outros.

Gordon e Gordon (2006), afirmam que os SADs ajudam os administradores nas tomadas de decisões dentro de uma organização. Uma das principais vantagens desse tipo de sistema é a capacidade de resposta mais rápida, ou seja, em fases decisórias dentro de uma empresa, os SADs auxiliam na melhor tomada de decisão.

Em relação ao SIE, Gordon e Gordon (2006) diz que são bem semelhantes aos SADs, onde a principal diferença é que esse tipo de sistema auxilia os executivos de grande escalão dentro de uma organização. Esse sistema tem como um de seus objetivos, auxiliar na tomada de decisão interna e externamente dentro de uma empresa.

Diante do exposto, pode-se concluir que existem diversos tipos de sistemas, e eles podem ser aplicados em diversas áreas. Cada sistema possui objetivos diferentes, dependendo a que sejam aplicados. Os sistemas são importantes para facilitar a realização de um conjunto de ações, e quando esses sistemas podem ser utilizados de qualquer lugar em que o usuário se encontra, essas tarefas podem ser realizadas com maior facilidade. É o caso dos sistemas *web*, tal que seus usuários podem acessar qualquer informação independentemente do lugar que se encontre.

2.2 TECNOLOGIA *WEB*

A tecnologia *web* fundamenta-se a partir do conceito da *internet*, que nada mais é que um conjunto de dispositivos interligados em rede em escala global, que trocam informações entre si. Essa tecnologia foi desenvolvida com o objetivo de formar um repositório do conhecimento humano, e se baseia no processo de funcionamento de armazenamento, recuperação e visualização de documentos eletrônicos (BERNERS-LEE et al., 1994 *apud* JUNIOR E VIDAL, 2006).

Desde a sua criação, até os dias atuais, a tecnologia *web* avança de maneira significativa, e o Brasil segue o ritmo acelerado ao crescimento dessa tecnologia. A evolução dessa tecnologia tem causado grande impacto nos sistemas de informações, tanto na sua manipulação, como também no seu desenvolvimento. Cada vez mais está havendo integração com outras tecnologias, agregando novos recursos e funcionalidades. Dessa forma, a *web* passou a ser não apenas um repositório de documentos eletrônicos, mais uma poderosa ferramenta de comunicação e universalização de acesso à informação.

2.3 LINGUAGEM DE PROGRAMAÇÃO PYTHON

Segundo Borges (2010, p.14), a linguagem de programação Python[1] foi criada em 1990 por Guido Van Rossum originada de uma linguagem de programação chamada *ABC*. A linguagem é considerada de alto nível e tem uma sintaxe simples, clara e elegante. Python é ainda uma linguagem expressiva, em que é mais fácil traduzir o raciocínio em um algoritmo e é distribuído com uma vasta biblioteca padrão, sendo também uma linguagem multiparadigma, ou seja, aceita que o programador utilize vários estilos de programação, suportando programação modular, funcional e Orientada a Objetos. A mesma por ser interpretada através de *Bytecode*, se transforma em uma linguagem de código portável, podendo assim compilar aplicações em diversas plataformas e sistemas operacionais.

O uso dessa linguagem de programação é frequentemente associado com grandes ganhos de produtividade e ainda, com a produção de programas de alta qualidade e de fácil manutenção.

Reis (2010) lista alguns recursos apresentados pela linguagem:

- Os conceitos fundamentais da linguagem são simples de entender.
- A sintaxe da linguagem é clara e fácil de aprender; o código produzido é normalmente curto e legível.
- Os tipos pré-definidos incluídos em Python são poderosos, e ainda assim simples de usar.
- A linguagem possui um interpretador de comandos interativo que permite aprender e testar rapidamente trechos de código.
- Python é expressivo, com abstrações de alto nível. Na grande maioria dos casos, um programa em Python será muito mais curto que seu correspondente escrito em outra linguagem. Isto também faz com o ciclo de desenvolvimento seja rápido e apresente potencial de defeitos reduzido – menos código, menos oportunidade para errar.
- Existe suporte para uma diversidade grande de bibliotecas externas. Pode-se fazer em Python qualquer tipo de programa, mesmo que utilize gráficos, funções matemáticas complexas, ou uma determinada base de dados SQL.
- É possível escrever extensões a Python em C e C++ quando é necessário desempenho máximo, ou quando for desejável fazer

[1] http://www.python.org.br

interface com alguma ferramenta que possua biblioteca apenas nestas linguagens.

- Python permite que o programa execute inalterado em múltiplas plataformas; em outras palavras, a sua aplicação feita para *Linux* normalmente funcionará sem problemas em *Windows* e em outros sistemas onde existir um interpretador Python.
- Python é pouco punitivo: em geral, `tudo pode' e há poucas restrições arbitrárias. Esta propriedade acaba por tornar prazeroso o aprendizado e uso da linguagem.
- Python é livre: além do interpretador ser distribuído como *software* livre (gratuito), pode ser usado para criar qualquer tipo de *software* — proprietário ou livre. O projeto e implementação da linguagem é discutido aberta e diariamente em uma lista de correio eletrônico, e qualquer um é bem-vindo para propor alterações por meio de um processo simples e pouco burocrático.

Na linguagem de programação Python dá-se grande ênfase à sua simplicidade e à flexibilidade de forma a maximizar sempre a produtividade do programador. No que se refere aos tipos e estruturas de dados, esta linguagem de desenvolvimento se apresenta na forma de uma tipagem dinâmica, porém forte. Os tipos das variáveis não precisam ser declarados pelo programador, como ocorre obrigatoriamente em diversas outras linguagens de tipagem estática, como por exemplo, a linguagem C. Embora seja simples, é também uma linguagem poderosa, podendo ser usada para administrar sistemas e desenvolver grandes projetos.

Com todas estas características benéficas se tornou uma linguagem muito popular dentre as grandes empresas de tecnologia e programadores, podemos citar a Google, Microsoft, Nokia, Nasa, dentre outras empresas de tecnologia.

2.4 *FRAMEWORK* DE DESENVOLVIMENTO DJANGO

Para se entender o que é o *Framework Django,* precisa-se definir primeiramente o que é *framework*. Mattsson (2000), afirma que um *framework* é uma arquitetura desenvolvida com o objetivo de atingir a máxima reutilização, representada como um conjunto de classes abstratas e concretas, com grande

potencial de especialização. Um *framework* compreende um conjunto de classes já implementadas em uma linguagem de programação específica, usadas para auxiliar o desenvolvimento de *software*.

Django[2] é um *framework* para desenvolvimento de aplicações escrito em Python, que por sua vez herda dessa linguagem de altíssimo nível agilidade no desenvolvimento de soluções e a facilidade na manutenção do código, visando assim acelerar o desenvolvimento, porém, mantendo o controle da aplicação nas mãos do desenvolvedor (DOCUMENTATION, 2015).

O Django é considerado um "*superframework*", pois é composto de vários *frameworks* (componentes) menores (SANTANA, 2010). O Django é um *framework* de desenvolvimento *web*, escrito totalmente em Python, com intuitos voltados para desenvolvimento ágil de *softwares*. Inicialmente foi desenvolvido por Jacob Kaplan-Moss, Adrian Holovaty e Simon Willison em 2003 como um CMS (abreviação do termo em inglês *Content Management System*, da qual significa sistema de gerenciamento de conteúdo) para um jornal do estado de Kansas – EUA, e devido ao seu grande potencial, foi transformado logo em seguida em um projeto *opensourse*.

A rapidez no desenvolvimento utilizando Django é modelada de acordo com o princípio DRY - *Dont Repeat Yourself*, evitando ao máximo códigos duplicados e gerando excelente ganho de agilidade no ciclo para desenvolvimento *web*, fazendo com que o desenvolvedor utilize de métodos que façam um reaproveitamento dos códigos. Classes e funções podem ser reutilizadas dentro de seu escopo no projeto, aplicando assim apenas algumas modificações em partes necessárias de seu código.

O Django utiliza o padrão MVC (*Model–view–controller*), é um padrão bastante aceito que visa a organização da aplicação, separando código, *layout* e modelos de forma organizada e padronizada. Belem (2013) descreve MVC como uma forma de estruturar seu projeto/aplicação de forma que a interface de interação (*view*) esteja separada do controle da informação em si (*models*), separação essa que é intermediada por uma outra camada controladora (*controllers*). Alguns aspectos positivos do Django:

[2] http://www.docs.djangobrasil.org/

- *Batteries Included*: o fato de vir com várias coisas de “fábrica”, ajuda bastante em seu desenvolvimento, poupa que o programador perca tempo fazendo muitas escolhas e configurações.
- Comunidade: Django, sem dúvida nenhuma tem a maior comunidade e aceitação entre os *frameworks web* Python.
- Estrutura: Django tem uma excelente estrutura base para projetos, onde pode-se organizar o projeto em apps tornando o projeto bem estruturado desde o início o que facilita a integração entre projetos e apps de terceiros.
- Simplicidade: a *startproject* do django gera poucos arquivos e pastas, o Mapeamento Objeto-Relacional (ORM) é um dos mais simples, o sistema de *templates* também, tudo gira em torno da simplicidade e do DRY.

Alguns aspectos negativos:

- Projetos muito pequenos: o Django perde um pouco da sua simplicidade, porque ele já considera que se vai precisar de um banco de dados relacional e de diversas outras configurações.
- Aprendizado: para ensinar Phyton e *web* Django não se sai bem, pelo mesmo fato de ter várias configurações que poderão ser desnecessárias, principalmente porque quando se ensina web com Python precisa-se escrever algumas funções para renderizar o texto e fazer apenas alguns cálculos simples.
- Projetos fora do padrão: para projetos que têm necessidade de coisas diferentes como um banco de dados não relacional, que não precise exibir páginas html, podem gerar improdutividade, tornando-se melhor usar outro *framework* mais flexível a escolhas, se for o caso.

O Django foi escolhido para integrar o projeto, pois com o mesmo obtém-se agilidade, organização nos projetos, pois tudo tem seu lugar definido, distribuídos entre as três camadas com suas responsabilidades que permite um trabalho bem “centrado” e modularizado.

2.5 LINGUAGEM DE MARCAÇÃO HTML E FOLHAS DE ESTILO CSS

HTML é sigla do termo em inglês *Hypertext Markup Language* que significa Linguagem de Marcação de Hipertexto e é essencialmente uma linguagem usada para definir o layout e a estrutura de páginas da Internet (STARK; JEPSON, 2012). Quando usuários utilizam sistemas *web*, as páginas visualizadas por eles são apenas documentos de texto no computador de outra pessoa.

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

Figura 1: Modelo de codificação HTML
Fonte: www.w3schools.com (Acesso em 13 de dezembro de 2015)

Na Figura 1, pode-se perceber que o HTML possui uma estrutura inicial padrão. Essa estrutura padrão foi criada para facilitar o desenvolvimento de documentos na linguagem. Além disso, as páginas em HTML são arquivos de texto bem simples, que podem ser criados e alterados em qualquer editor de textos, como o Bloco de Notas do Windows. O texto em uma página *web* típica é encapsulado em *tags* HTML, que informam ao navegador (*software* utilizado para acessar a *Internet*) a estrutura do documento que deve ser lido para o conteúdo a ser mostrado. Dessa maneira, o navegador sabe como exibir as informações de uma maneira que faça sentido. Os documentos HTML são divididos em partes, onde cada parte contém uma informação específica.

> "Os documentos HTML são armazenados como simples arquivos de texto com extensão .html ou .htm. É possível abri-los em qualquer editor de texto. Exibindo o código-fonte HTML é possível entender que todas as páginas da *Internet* se reduzem no fundo a documentos de textos HTML que definem seu *layout* e estrutura." (MORRISON, 2000, P. 13 e 14)

CSS é sigla do termo em inglês Cascading Style Sheets, é uma linguagem de folhas de estilos utilizadas para definir a apresentação visual de um documento em uma linguagem de marcação HTML ou XML. Seu principal benefício é prover a separação entre o formato e o conteúdo de um documento. Segundo Eis e Ferreira (2012), CSS é a linguagem responsável por controlar o visual da informação exibida pelo HTML. A informação é acessada por diferentes meios que façam uso das tecnologias *web*, e o CSS é o responsável por formatar essas informações para que sejam consumidas independentes da maneira que sejam acessadas, adaptando para cada meio de acesso. Em 1995, foi lançada a recomendação oficial pelo W3C do CSS 1.0. Embora a especificação do CSS tenha sido lançada em 1995, os *Browsers* levaram mais três anos para ter suporte completo ao recurso. Atualmente se encontra na versão 3.

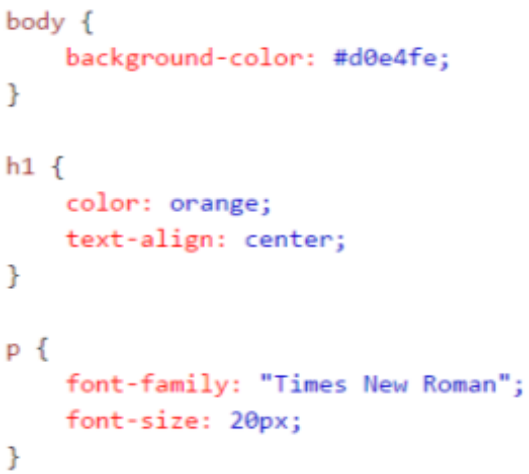

Figura 2: Modelo de codificação CSS
Fonte: w3schools (Acesso em 19 de dezembro de 2015)

Ao invés de colocar a formatação dentro do arquivo HTML o desenvolvedor cria uma ligação para uma página que contém os estilos em um CSS, procedendo de forma idêntica para todas as páginas de um portal, ou seja, qualquer página do portal chama esse arquivo CSS e assim todo o portal mantém um padrão de cor e tamanho. Quando é necessário alterar a aparência do portal, basta modificar esse arquivo, reduzindo drasticamente o esforço e mantendo um padrão (FREEMAN, 2007). A Figura 2 mostra um exemplo de um arquivo CSS.

2.6 JAVASCRIPT

Segundo Santos (2009), *JavaScript* é uma linguagem de programação que possui instruções que lhe possibilita adicionar um novo nível de interatividade e função as páginas *web*, ou seja, é responsável pelo comportamento das páginas HTML. Além disso, *script* é uma sequência de instruções feita com linguagem de programação, e que são interpretados ou executados por um *software*.

Essa camada é responsável por todos os comportamentos da página HTML. É nessa camada que a aplicação ganha maior dinamismo e interatividade com seus usuários, tratando todos os eventos do sistema, como: cliques em qualquer parte da página *web*; ocultar elementos específicos da página HTML; mudar características do CSS da página, entre outros. O *JavaScript* é considerado o principal responsável por manipular o HTML e o CSS. Sobre *JavaScript*, Eis e Ferreira (2012) afirmam que essa tecnologia, até hoje, é a principal responsável pela interatividade e dinamismo das páginas *web*, ou seja, é a principal responsável pela comunicação entre o usuário e a aplicação.

De acordo com Stark e Jepson (2012), *JavaScript* é uma linguagem de *script* que pode ser adicionada a uma página HTML para torna-la mais interativa e conveniente para o usuário. Por exemplo, com um código *JavaScript*, é possível inspecionar valores digitados em um formulário para garantir que sejam valores válidos. Através do *JavaScript* também é possível mostrar ou esconder elementos de uma página, dependendo de onde o usuário clicar. Essa linguagem de *script* pode até contatar o servidor *web* para executar alterações no banco de dados sem atualizar a página *web* atual.

Com o *JavaScript* é possível definir se os elementos serão arrastados, dimensionados, rotacionados, reformatados, etc. O *JavaScript* controla tudo isso manipulando as características dos elementos CSS. Resumidamente, o *JavaScript* controla os valores definidos pelo CSS e manipula estas propriedades, (EIS; FERREIRA, 2012).

O código *JavaScript* pode ser definido dentro de um mesmo arquivo HTML, sendo que o mesmo deve estar dentro da *tag* <script>, ou está contido em um arquivo *JavaScript* próprio, com a extensão .js, que especifica que o arquivo é um código *JavaScript*. Na Figura abaixo é apresentado um exemplo simples de código

JavaScript, sendo que o código faz com que apareça na tela do navegador, uma caixa de mensagem com um texto pré-definido.

```
<script >
    alert ("Bem-vindo");
</ script >
```

Figura 3: Código JavaScript

2.7 BANCO DE DADOS MYSQL

O MySQL[3] é um sistema gerenciador de Banco que utiliza a linguagem SQL (Linguagem de Consulta Estruturada, do inglês *Structured Query Language*) de código-fonte aberto (sob licença GPL), o que significa que qualquer um pode estudá-lo ou alterá-lo conforme a necessidade. Este banco de dados é conhecido por sua facilidade de uso, flexibilidade e confiabilidade. Outro ponto positivo desta ferramenta é por a mesma ser multiplataforma, ou seja, pode atuar em diversas plataforma e sistemas operacionais como Linux, FreeBSD (e outros sistemas baseados em Unix), Windows e Mac OS X (DOCUMENTATION, 2015).

Milani (2007, p. 24) afirma que atualmente o MySQL tem sido o banco de dados *Open-Source* mais utilizado em Aplicações *web*, pelo fato dessas aplicações demandarem rápido acesso ao banco para geração de páginas, seja qual for a linguagem de programação que faça conexão com o banco.

2.8 *FRAMEWORK* DE FRONT-END BOOTSTRAP

O Bootstrap é um *framework* de código aberto desenvolvido por Mark Otto e Jacob Thornton que a lançaram enquanto eram funcionários do *Twitter*, pela necessidade de padronizar os conjuntos de ferramentas de *front-end* de engenheiros em toda a empresa (SPURLOCK, 2013).

[3] http://www.mysql.com/

Com o *Bootstrap* o desenvolvedor não precisa definir a partir do zero os elementos básicos da *interface* como botões. Nele estão à disposição vários elementos de *interface* de uso comum. *Bootstrap* é um kit de ferramentas com as convenções padrão de classes com documentação limpa e prática para dar vida a códigos que estão prontos para serem usados e personalizados de acordo com as necessidades dos desenvolvedores. Não é uma solução mágica para resolver o problema de reutilização de elementos da *interface*, mas é um bom início (MAGNO, 2013).

Pelas facilidades que apresenta em reutilização de componentes prontos o *bootstrap* vem sendo utilizado cada vez mais pelos desenvolvedores, pois permite uma maior agilidade na hora de montar as *interfaces* dos seus sistemas *web*. Não é preciso criar e dar estilo a cada componente da sua interface. Basta associar os elementos das páginas HTML às classes que existem prontas no *bootstrap* para reutilizar-se botões, menus, barras de navegação, formulários e outros componentes comumente utilizados em *interfaces web*.

2.9 ODONTOGRAMA

Odontograma é uma espécie de formulário (impresso ou digital) utilizado por profissionais de Odontologia, que contém a representação gráfica dos dentes do paciente. Geralmente é na primeira consulta, que é preenchida a situação em que se encontra cada elemento dentário, ou seja, o dentista faz anotações das características de cada elemento dentário. Para que seja feita essas anotações, utiliza-se um desenho ou um esquema de cada dente sendo marcadas as variações anatômicas e procedimentos que serão ou foram realizados, assim como a ausência de algum dente. Segundo Rubira e Rodrigues (1988), o odontograma é um diagrama gráfico onde estão representados os dentes permanentes e decíduos, possuindo um código de preenchimento pré-estabelecido, seguindo um tipo de notação dental.

A Figura 4 apresenta um modelo de odontograma impresso que é utilizado no sistema odontológico, com os dados do paciente e uma representação gráfica de cada elemento dentário, onde é marcada a situação de cada dente, e um campo para anotações que forem necessárias.

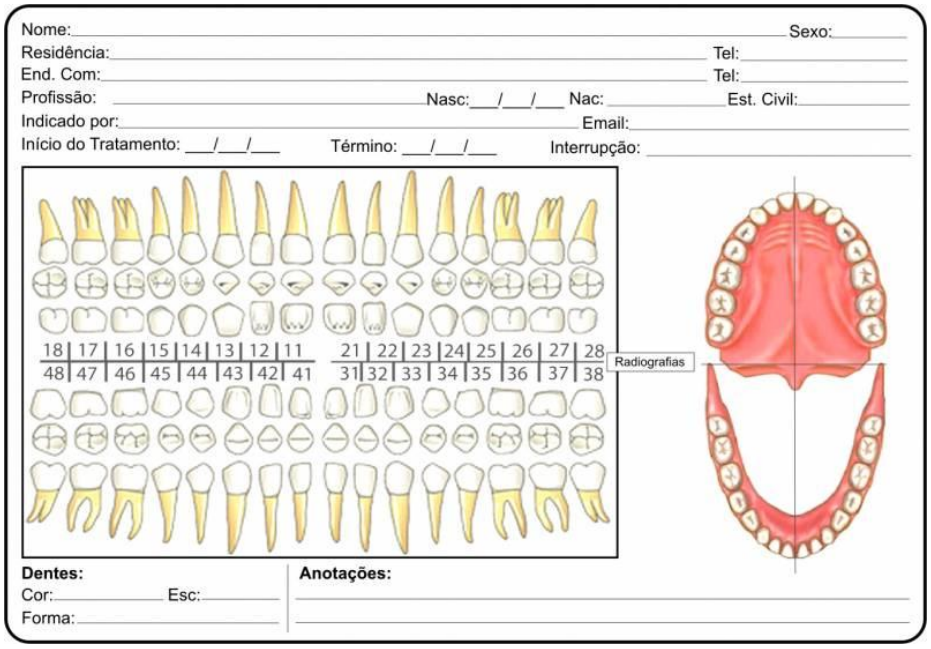

Figura 4: Modelo de Odontograma impresso.
Fonte: www.odontokit.com.br (Acesso em 10 de dezembro de 2015)

2.10 METODOLOGIAS DE DESENVOLVIMENTO

O desenvolvimento de *Software*, na década de 70, ficou conhecido pela Crise do Software (PRESSMAN, 2006), pois nesse período a produção de *software* era feita de forma desorganizada, desestruturada, sem planejamento, sem produção de documentação e a análise do projeto não utilizava métodos no seu desenvolvimento. Com isso, prazo e custo não correspondiam a real necessidade.

"A Engenharia de *Software* utiliza o Processo de Desenvolvimento, que consiste na criação de documentos, artefatos e marcos, capazes de representar o contexto do *software*, levando em consideração recursos, ferramentas, prazos, restrições, e outros aspectos que envolvem o desenvolvimento de um produto, para no final produzir software de qualidade." (PRESSMAN, 2006).

Ainda no contexto de engenharia de *software*, temos as metodologias tradicionais e as ágeis. Segundo Pressman (2006), as metodologias tradicionais surgiram em um cenário de construção de *software* muito diferente do atual. A sequência de tarefas para desenvolvimento era realizada em "terminais burros" e *mainframes*, onde o custo para correção de erros ou qualquer outra alteração era muito elevado. O acesso aos computadores era muito limitado e não haviam ferramentas de apoio à construção de *software*, com isso para minimizar os

problemas durante o desenvolvimento, necessitava-se planejar e projetar antes. Esse foi um dos fatores primordiais para a existência de um processo de gerenciamento de *software*, baseado em documentação detalhada, planejamento extenso e etapas bem delimitadas. Exemplos: Cascata, espiral:

- Cascata: Conforme Pressman (2006), o modelo em cascara sugere uma abordagem linear e sequencial de todas as atividades envolvidas no desenvolvimento, e por se tratar de uma sequencia de eventos, esse modelo também ficou conhecido como “ciclo de vida”.
- Espiral: Segundo Pádua (2000), no modelo espiral o produto é desenvolvido em uma série de iterações. Cada nova iteração corresponde a uma volta na espiral. Isto permite construir produtos em prazos curtos, com novas características e recursos que são agregados na medida em que a experiência descobre sua necessidade.

Metodologias ágeis de gerenciamento de projetos apareceram com contraposto às chamadas metodologias pesadas – metodologias tradicionalistas – como uma alternativa para se adaptar ao contexto do mercado atual, que exige resultados cada vez mais rápidos para atender às necessidades dos clientes, sob condições de constantes mudanças e incertezas. Pesquisas realizadas na Pontifícia Universidade Católica do Rio Grande do Sul em 2005 demonstram que somente 16,2% dos quase 8380 projetos usados por base da pesquisa, foram entregues dentro dos prazos, custos e com todas as funcionalidades especificadas (SOUZA, 2008).

De Acordo com Cockburn (2002), os métodos ágeis são adaptativos e não previsíveis, como é o caso das metodologias tradicionais. Eles se adaptam aos novos fatores que surgem durante o desenvolvimento do projeto ao invés de verificar antecipadamente todos os impedimentos que podem acontecer durante a construção do produto. O problema em si não está nas mudanças, pois estas estão sempre sujeitas a ocorrer em quaisquer dos âmbitos, o problema está na maneira que cada uma das metodologias trata essas mudanças. Como recebem, avaliam e respondem às mesmas. Exemplos: *Scrum*, *eXtreme Programming* (XP), Cristal:

- *Cristal*: Segundo Pressman (2006), a metodologia cristal foi criado por Alistair Cockburn e Jim Highsmith, visando conseguir elaborar uma abordagem de desenvolvimento que priorizasse a adaptabilidade durante o que Cockburn (2002) caracteriza como um jogo de comunicação cooperativa e com recursos

limitados, tendo como primeiro objetivo entregar *software* útil em funcionamento e como segundo preparar-se para o jogo seguinte.

- *eXtreme Programming* (XP): Segundo Teles (2004), a XP é um processo de desenvolvimento de *software* apropriado para os seguintes projetos: requisitos vagos que e mudam com frequência; Desenvolvimento de sistemas orientados a objeto; Equipes pequenas; e desenvolvimento incremental.
- *Scrum*: De acordo com Sabbagh (2013), Scrum é um framework Ágil, simples e leve, utilizado para a gestão do desenvolvimento de produtos complexos imersos em ambientes complexos. O autor afirma ainda, que a metodologia de desenvolvimento adotada pelo Scrum faz com que os desenvolvedores ou a equipe de desenvolvimento entregue partes do projeto com frequência, reduzindo assim, os riscos do projeto. Segundo Sabbagh (2013) o uso do Scrum traz benefícios que incluem:
    - Entregas frequentes de retorno ao investimento dos clientes;
    - Redução dos riscos do projeto;
    - Maior qualidade no produto gerado;
    - Mudanças utilizadas como vantagem competitiva;
    - Visibilidade do progresso do projeto;
    - Redução do desperdício;
    - Aumento de produtividade.

2.11 LINGUAGEM DE MODELAGEM UNIFICADA (UNIFIED MODELING LANGUAGE – UML)

Durante o desenvolvimento de um projeto, é aconselhável e de fundamental importância, realizar a análise e coleta de requisitos do projeto. Segundo Pádua (2003), o valor de um produto vem de suas características, e os requisitos são as características que definem os critérios de aceitação de um produto. O processo de análise de requisitos é um levantamento das funcionalidades que o sistema deve atender e as tarefas que o mesmo irá realizar, entendendo e colocando no papel, o que uma aplicação destina-se a fazer depois de construída.

Durante a análise de requisitos, é de fundamental importância a construção de gráficos e diagramas que representem as funcionalidades da aplicação. Isso é

feito através da Linguagem de Modelagem Unificada (*Unified Modeling Language – UML*) que segundo Braude (2005) é uma notação gráfica para expressar projetos orientados a objetos. Ela faz uso de diagramas e gráficos para representar as funcionalidades do sistema. Existem diversos diagramas que podem fazer representação do sistema, alguns exemplos são: diagrama de classe; diagrama de caso de uso; diagrama de sequência, entre outros.

Conforme Guedes (2011), UML é uma linguagem visual utilizada para modelar *softwares* baseados no paradigma de orientação a objetos, ou seja, é uma linguagem de modelagem de propósito geral que pode ser aplicada a todos os domínios de aplicação. Além disso, essa linguagem tornou-se, nos últimos anos, a linguagem padrão de modelagem adotada internacionalmente pela indústria de engenharia de *software*.

UML não é uma linguagem de programação, e sim uma linguagem de modelagem, uma notação, cujo objetivo é auxiliar os engenheiros de *software* e desenvolvedores a definirem as características do sistema. Por mais simples que seja, todo e qualquer sistema deve ser modelado antes de se iniciar sua implementação, pois todo sistema de informação costuma mudar, o que acaba facilitando a manutenção do sistema, caso exista uma modelagem da aplicação, (GUEDES, 2011).

3 ESPECIFICAÇÕES DO SISTEMA / IMPLEMENTAÇÃO

Esse capítulo descreve o protótipo do Sistema de Auxílio ao Serviço Odontológico da UFPI desenvolvido denominado de Easy Oral Health e são apresentados os requisitos funcionais, não funcionais e os principais diagramas que auxiliaram no desenvolvimento do protótipo, como os diagramas de casos de uso, de classes, entre outros. Além disso, são discutidos sobre o levantamento de requisitos, objetivando um melhor entendimento para o reconhecimento dos detalhes do sistema.

Para conceber a implementação da aplicação foram utilizados vários artifícios. Entre eles estão o sistema operacional Ubuntu na versão 14.04 LTS, executado em uma máquina virtual utilizando a aplicação VirtualBox, versão 5.0.10, e para facultar a execução de várias aplicações simultaneamente utilizou-se um notebook com processador intel core I3, memória RAM 6gb e um disco de 500gb.

Para o desenvolvimento do projeto fez-se uso da linguagem de programação *Python* versão 3.4 juntamente com o *Framework* Django versão 1.9. A preferência por essa linguagem foi devido ser uma linguagem de alto nível possuindo uma de sintaxe simples e clara, sendo uma linguagem de grande crescimento nos últimos anos. A escolha do uso do Django se deve ao fato de ser um *framework* escrito integralmente em *Python*, além de possuir recursos atrativos que aumentam significativamente a produtividade, também possui uma gama de bibliotecas portáveis e uma documentação que contém todos os elementos necessários para o entendimento da ferramenta e suas funções.

A metodologia de desenvolvimento utilizada foi o *Scrum*, pois essa metodologia possui entre suas características o acompanhamento do desenvolvimento de perto por parte dos usuários, sempre analisando o que está sendo feito e validando.

Entretanto, para que o usuário possa conseguir utilizar as funcionalidades implementadas, faz-se necessário criar uma interface intuitiva que possibilite a interação entre o usuário e o sistema. Para o desenvolvimento da interface do *Easy Oral Health* foi utilizado o *framework Bootstrap* contendo diversos princípios que compõem páginas *web,* além de dispor de códigos que implementam técnicas de

responsividade, permitindo a adaptação da página em uma grande variedade de dispositivos, possibilitando uma navegabilidade mais agradável.

Quadro 1- Etapas do desenvolvimento

Etapa	**Descrição**
1° Etapa	Levantamento de Requisitos (entrevistas, pesquisas).
2° Etapa	Análise e Design da Aplicação (diagramas UML).
3° Etapa	Modelagem dos dados (criação do diagrama de entidade e relacionamento).
4° Etapa	Desenvolvimento (criação dos formulários, telas, funcionalidades e etc.).
5° Etapa	Testes (teste das funcionalidades da aplicação).

O desenvolvimento do projeto passou por diversas etapas, essas etapas podem ser visualizadas no Quadro 1, onde se iniciou pela análise e especificação necessária para levantar as reais necessidades do cliente em relação às funcionalidades do sistema. Seguido pela Análise e *Design* utilizados para proporcionar uma visão geral da arquitetura do sistema, através da modelagem da solução, com o uso de diagramas UML. Após essas fases, foi realizada a modelagem de dados, que contém o diagrama de entidade relacionamento. Esse diagrama tem como finalidade descrever, de maneira conceitual, os dados utilizados no desenvolvimento do sistema. Logo em seguida foram elaborados os formulários e telas do sistema juntamente com as funcionalidades previstas na fase de levantamento de requisitos. Para finalizar, uma série de testes foram realizados na aplicação, visando a se todas as funcionalidades estavam de acordo com os requisitos.

3.1 REQUISITOS DO SISTEMA

Durante o processo de levantamento de requisitos do sistema, foram realizadas diversas reuniões e entrevistas com pessoas ligadas ao setor de odontologia do CSHNB, em cada encontro foram tratados assuntos específicos para possibilitar a modelagem do sistema.

Posteriormente, foram formulados os requisitos funcionais e não funcionais do sistema. O Quadro 2 contém detalhes sobre os requisitos funcionais identificados por um código único, com suas respectivas descrições e suas dependências em casos específicos. As dependências são interligadas exclusivamente pelo código

único do requisito a qual é dependente, por exemplo, o requisito funcional RF002 depende do requisito funcional RF001.

Quadro 2- Requisitos funcionais

Identificador	**Descrição**	**Dependência**
RF001	O sistema deverá ter um usuário administrador padrão do sistema.	
RF002	O sistema deverá possibilitar o gerenciamento de usuário e grupos de usuário, assim como suas permissões de acesso.	RF001
RF003	O sistema dará permissões somente ao administrador geral do sistema para cadastrar novos usuários.	RF002
RF004	O sistema possibilitará somente a criação de dois tipos de usuário, Administrador e Dentista.	RF003
RF005	Os níveis de acesso aos recursos do sistema serão de acordo com os tipos de usuários.	RF004
RF006	O sistema deverá possuir a autenticação do usuário através de uma página de Login.	RF002, RF005
RF007	O sistema deverá possibilitar o gerenciamento dos seguintes itens básicos: Cursos, Dentes, Divisão dos dentes, Status dos dentes, Procedimento dentários, Procedimento Bucal, Pacientes, Dentistas e Odontograma.	RF006
RF008	O usuário dentista poderá realizar consultas odontológica, bem como inserir informações a respeito de cada paciente.	RF005, RF006
RF009	O sistema deverá possuir três tipos de paciente: Estudante, Funcionário e Dependente.	
RF010	O sistema deverá usar critérios de distinção entre pacientes que são estudantes e pacientes que são funcionários, objetivando o registro de dependentes.	RF009
RF011	Paciente do tipo Funcionário terá direito a dependentes.	RF010
RF012	Os pacientes cadastrados no sistema terão direito a todos os procedimentos odontológicos disponíveis, sem distinção.	
RF013	O sistema deverá possuir dois tipos de procedimentos: Dentário e Bucal.	
RF014	O sistema deverá possibilitar ao dentista criar um Odontograma para cada paciente, objetivando manter as informações atualizada da arcada dentária do paciente.	RF008

O Quadro 3 apresenta os requisitos não funcionais do sistema, identificados por um código único, seguido de suas descrições e sua respectiva categoria pela qual o requisito se enquadra, por exemplo, o requisito identificado com o código RNF001 está na categoria de confiabilidade, robustez e implementação e sua descrição diz que o sistema deve ser implementado usando técnicas de modularização.

Quadro 3 - Requisitos não funcionais

Identificador	Descrição	Categoria
RNF001	O Sistema deve ser implementado usando técnicas de modularização.	Confiabilidade – Robustez – Implementação.
RNF002	O sistema deverá ser implementando usando técnicas de responsividade.	Portabilidade – Implementação.
RNF003	O sistema deverá ser implementado usando o princípio Don't Repeat YourSelf (DRY).	Velocidade.
RNF004	O sistema deverá ser instalado e configurado em um equipamento servidor de alto poder de processamento.	Infra-Estrutura.
RNF005	O sistema deverá suportar no mínimo 100 requisições simultâneas por segundo.	Eficiência.
RNF006	O sistema deverá ter disponibilidade de no mínimo 98% do tempo.	Confiabilidade.
RNF007	O sistema deverá ser implementado usando a linguagem de programação Python juntamente com um *framework* de desenvolvimento Django.	Implementação – Padrões.
RNF008	O sistema deverá recuperar dados apenas para usuários autorizados a acessar tal informação.	Segurança de acesso.
RNF009	O sistema deverá apresentar telas intuitivas de fácil aprendizagem.	Facilidade de uso.
RNF0010	O sistema deverá responder em um intervalo de tempo de no máximo 100ms uma requisição de usuário.	Tempo de execução.
RNF0011	O sistema deverá se comunicar com um Sistema Gerenciador de Baco de Dados Relacional (SGBDR) MySQL.	Interoperabilidade – Manutenibilidade.
RNF0012	O sistema, assim como seus aplicativos deverão rodar de forma isolada e independente de qualquer outra aplicação externa através do *software VirtualEnv.*	Segurança de acesso – Confiabilidade.

3.2 DIAGRAMAS DE CASO DE USO

Guedes (2011, p.52) afirma que:

> O diagrama de casos de uso é de grande auxílio para a identificação e compreensão dos requisitos do sistema, ajudando a especificar, visualizar e documentar as características, funções, serviços do sistema desejados pelo usuário. O diagrama de casos de uso tenta identificar os tipos de usuários que irão interagir com o sistema, quais papéis esses usuários irão assumir e quais funções um usuário específico poderá requisitar.

A partir das informações sobre diagrama de casos de uso pode-se entendê-lo como um meio que fornece uma visão geral do funcionamento do sistema e os usuários que estão envolvidos em cada funcionalidade.

O diagrama de casos de uso da Figura 5 descreve as ações básicas do sistema. O Usuário Administrador é considerado o ator principal nas ações das tarefas administrativas, ele é o responsável pelo gerenciamento de usuários, grupos

de usuários e suas permissões de acesso específicas, como também nas tarefas básicas como o gerenciamento de cursos, dentes, divisão de dentes, status dos dentes, status do procedimento, procedimento dentário, procedimento bucal e também o cadastro de dentistas. Já os atores Usuário dentista, pode gerenciar seus dados específicos como nome, *e-mail,* senha e outros e também os dados dos pacientes e suas informações (Nome, endereço, informações dentarias e os procedimentos concluídos e a serem feitos, bem como o odontograma.

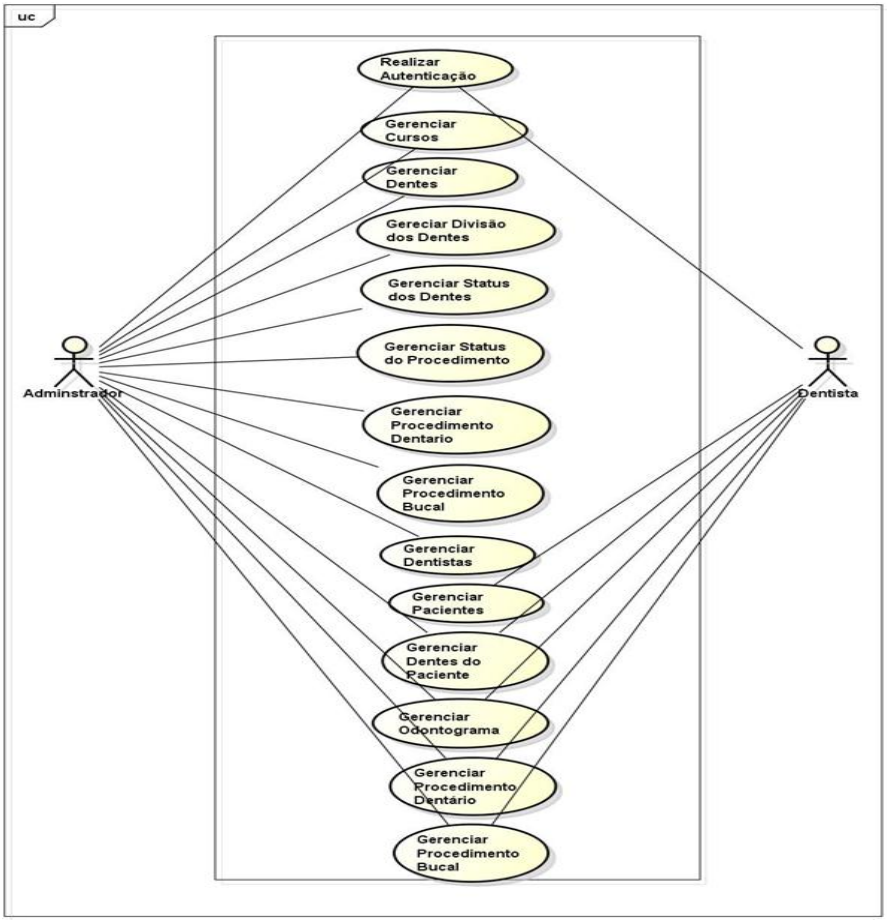

Figura 5: Diagrama de caso de Uso

3.3 DIAGRAMA DE BANCO DE DADOS

Segundo Heuser (2009), o diagrama de banco de dados é uma descrição real do banco de dados da aplicação, ou seja, ele é o responsável por detalhar quais tabelas o banco irá possuir, e quais os atributos de cada tabela. Dessa maneira, é possível os desenvolvedores terem uma visão mais ampla em relação ao sistema.

A Figura 6 representa o diagrama de banco de dados que foi desenvolvido durante esse projeto. Nele estão contidas todas as tabelas pertencentes ao banco de dados da aplicação, e todos os atributos de cada tabela. Lembrando que esse diagrama foi desenvolvido de acordo com o levantamento de requisitos (Quadro 1) do sistema, ou seja, esse diagrama foi elaborado com o intuito de atender e satisfazer as reais necessidades do sistema.

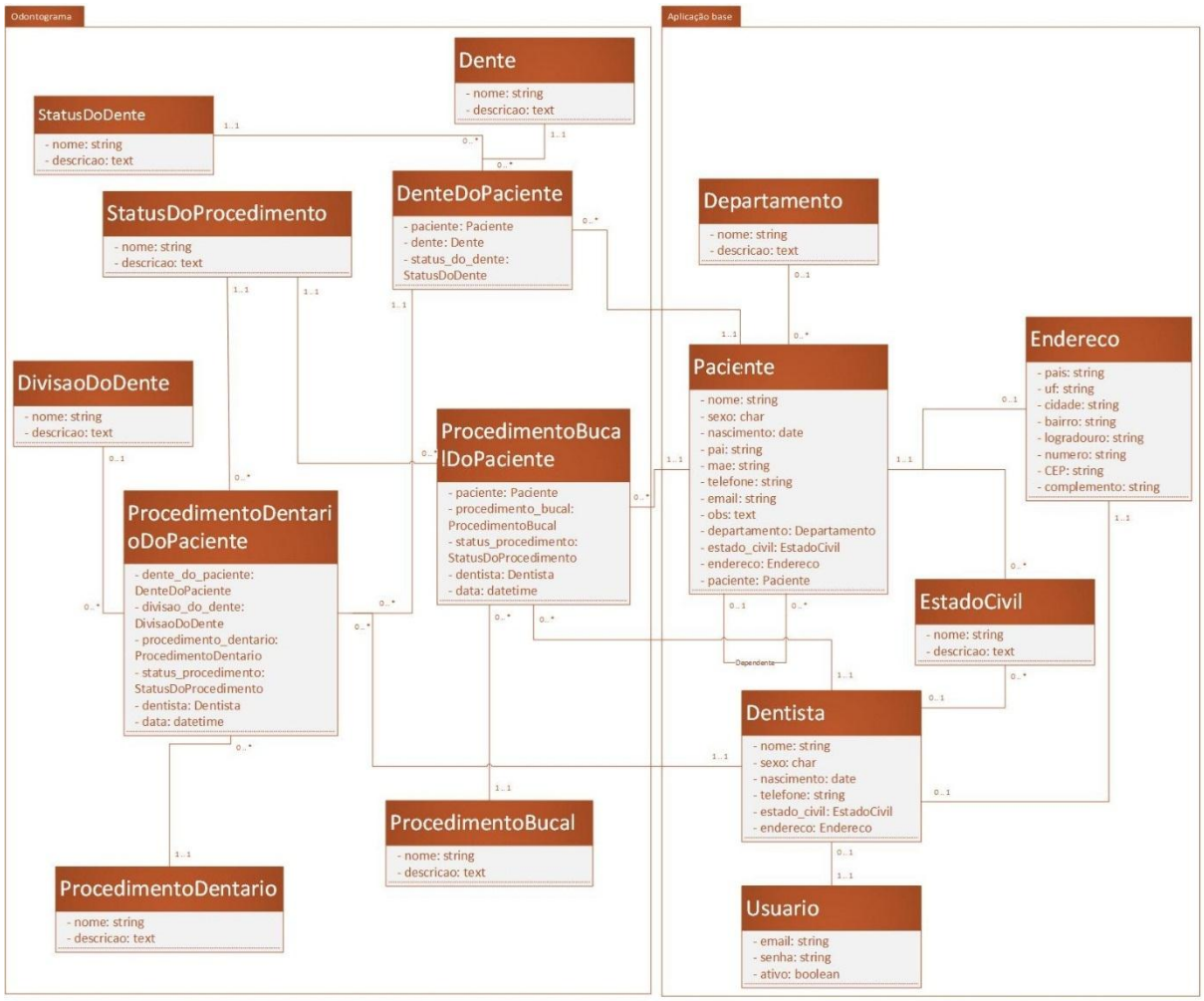

Figura 6: Diagrama de Banco De Dados

3.4 TELA INICIAL

A Figura 7 ilustra a tela inicial do sistema, que se encontra de maneira bastante simplificada no intuito de facilitar o acesso aos serviços oferecidos. A barra de navegação superior oferece navegabilidade completa através dos menus que são responsáveis pelo redirecionamento para o serviço desejado.

Figura 7: Tela Inicial

Há três menus principais localizados de forma centralizada na tela com as seguintes denominações: “Inicio”, é responsável pelo redirecionamento para página principal; “Cadastros”, é responsável por exibir um conjunto de sub-menu onde podemos efetuar o cadastro de Cursos, Dentes, Divisão de dentes, Status do dente, Status do procedimento, Pacientes e Dentista, podendo mais visualizar cada cadastro e altera-los; e no menu “Procedimentos” podemos cadastrar, visualizar e editar os procedimentos realizados e a serem realizados no paciente, contando que o procedimento pode ser na boca ou nos dentes.

Toda a interface *frontend* do Easy Oral Health foi desenvolvida utilizando o *framework bootstrap,* proporcionando uma interação-humano-computador (IHC) agradável. O *bootstrap* possui classes com folhas de estilização de páginas *Cascading Style Sheets* (CSS), que além de fazer a apresentação dos documentos, também oferece recursos de adaptação do conteúdo de acordo com o tamanho da tela, permitindo assim uma ótima navegabilidade em boa parte dos dispositivos existentes com acesso à *internet*.

A Figura 8 mostra a tela inicial utilizando a responsividade do sistema, destacando a visualização dos menus na simulação realizada em um dispositivo movel *android.*

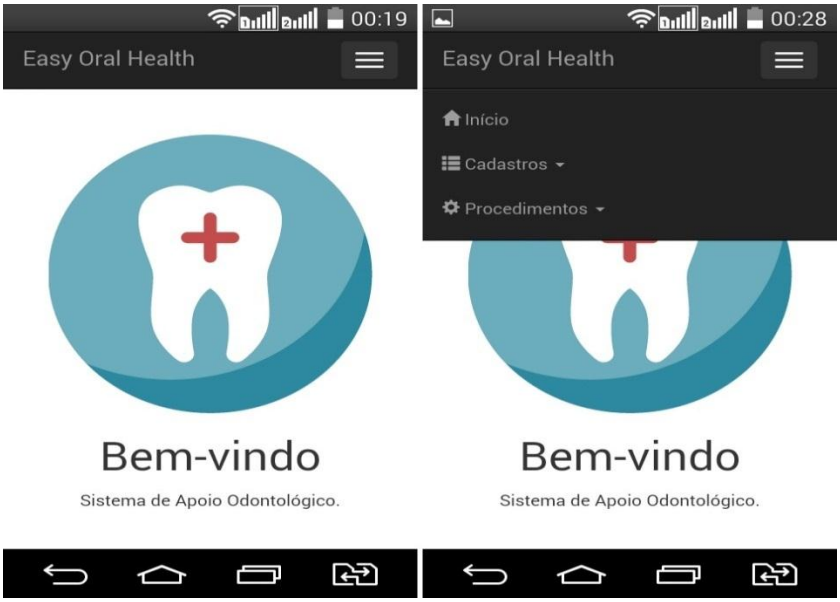

Figura 8: Tela Inicial Responsiva

3.5 AUTENTICAÇÃO

Para acessar as telas administrativas o usuário necessita autenticar-se no sistema. A tela de autenticação de usuários do *Easy Oral Health* pode ser visualizada na Figura 9, sendo necessário apenas o preenchimento do usuário e senha. Para que um usuário possa ter acesso ao sistema o administrador deverá ter cadastrado o mesmo antecipadamente.

Figura 9: Tela de Autenticação

3.6 CADASTROS

As tarefas gerenciáveis de criar, listar, alterar e excluir do Easy Oral Health seguem o mesmo padrão, dessa forma são demonstradas apenas algumas telas de cadastro e listagem do módulo administrativo.

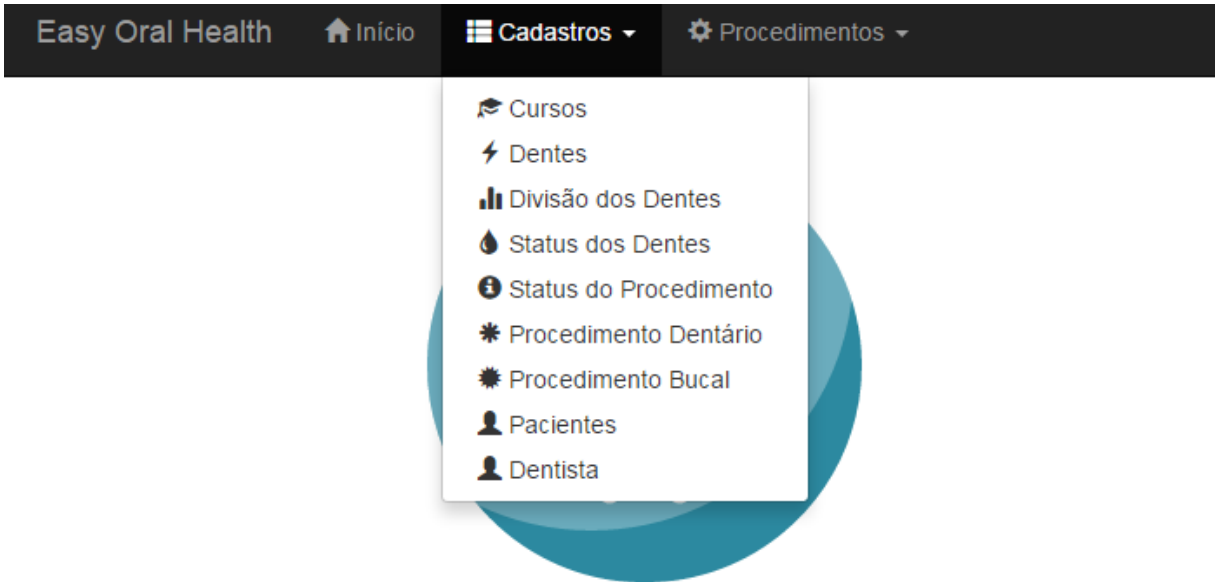

Figura 10: Menu de cadastros

Na Figura 10 podemos ver o menu com todas as opções de cadastro, existentes no sistema, já na Figura 11 exemplifica como é feito o cadastro dos pacientes no sistema. Existe um formulário com alguns campos a serem preenchidos (os campos fazem referência aos atributos modelados no diagrama de banco de dados ilustrado na Figura 6). Logo acima possui um botão para cancelar o cadastro.

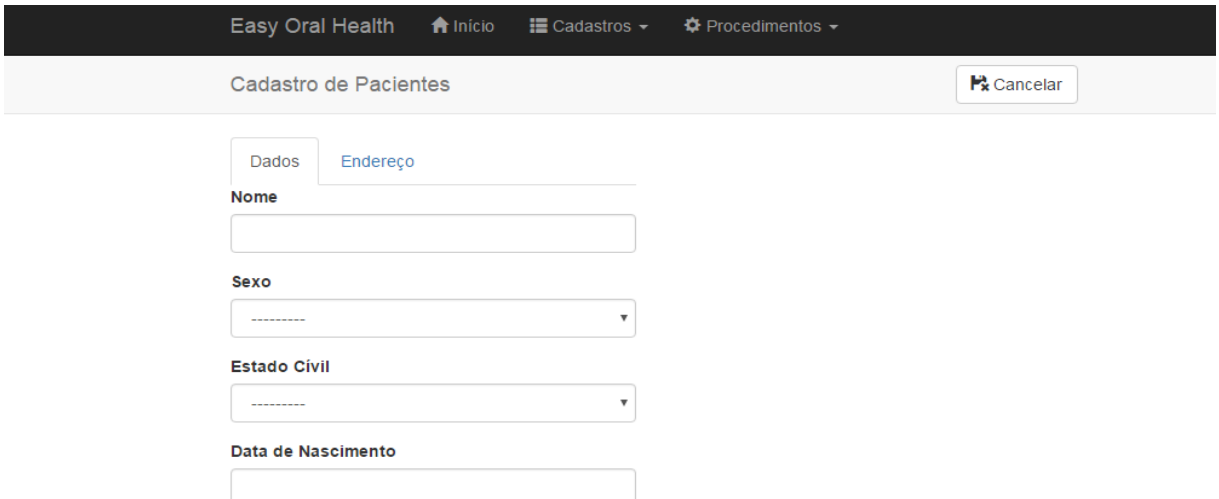

Figura 11: Formulário de cadastro

3.7 LISTAGENS

O Easy Oral Health é capaz de recuperar dados de forma a exibi-los ao usuário, necessitando apenas que existam dados disponíveis. A Figura 12 mostra algumas listagens de dados previamente cadastrados, podemos visualizar o cadastro dos pacientes e seus dependentes. Os dependentes aparecem abaixo no cadastro, destacados por uma barra da cor verde.

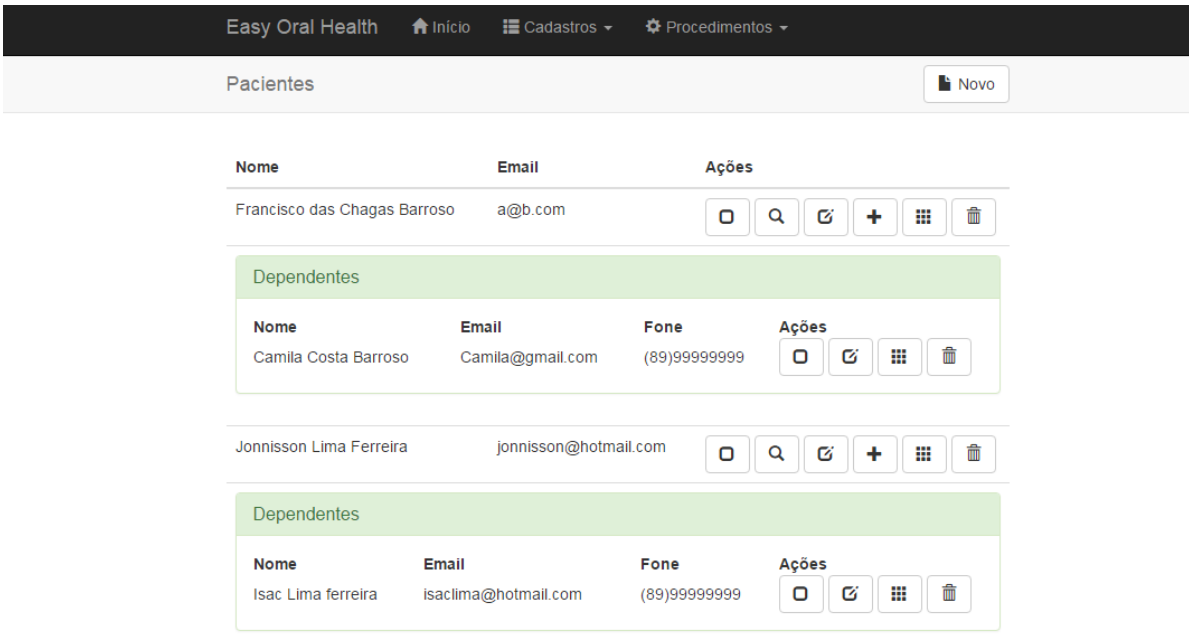

Figura 12: Listagem de Pacientes

3.8 ALTERAÇÃO DE DADOS

A figura 13 exemplifica a edição de um cadastro de pacientes já cadastrados no banco de dados. Ao entrar na tela de edição, será mostrado ao usuário cada campo preenchido da maneira que foi cadastrado no banco de dados anteriormente, facilitando assim a edição das informações necessárias. No canto superior direito possui um botão “Cancelar”, para que o usuário possa interromper a ação quando desejar.

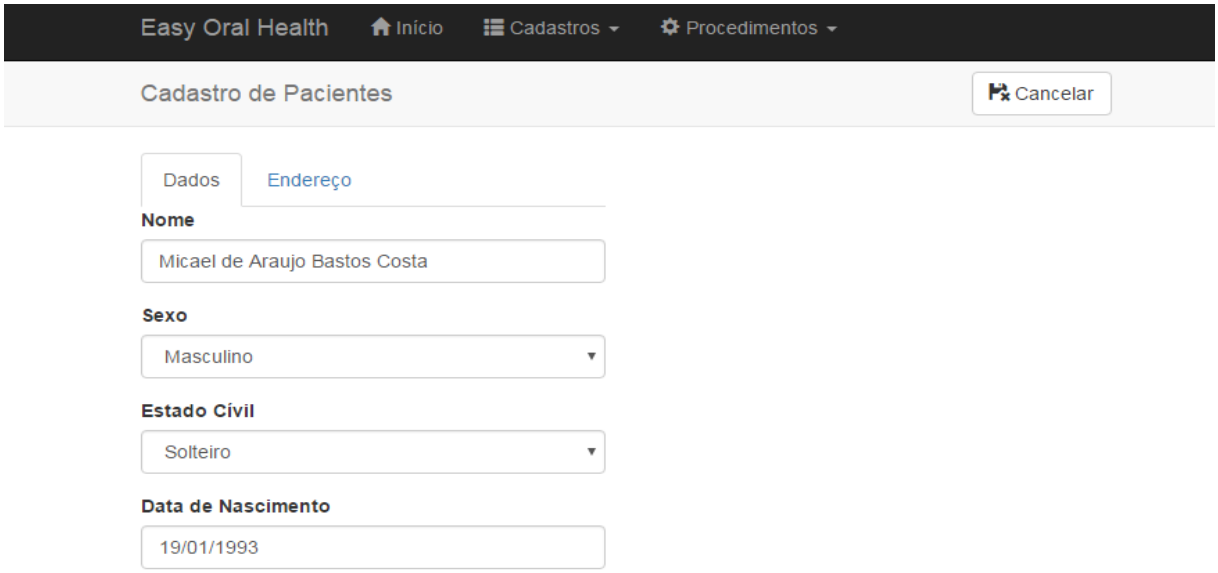

Figura 13: Alteração de Paciente

3.9 ODONTOGRAMA

O Odontograma é uma representação gráfica da arcada dentaria do paciente que ajuda ao odontólogo no acompanhamento do tratamento. Cada dente é dividido em 5 partes onde cada uma dessas parte podem ser marcadas com cores diferente de acordo com os procedimentos já realizado ou os que ainda não foram. É possível mostrar que o dente está ausente ou foi extraído. A Figura 14 mostra o modelo de Odontograma utilizado no sistema e a legenda de cores correspondentes para cada procedimento.

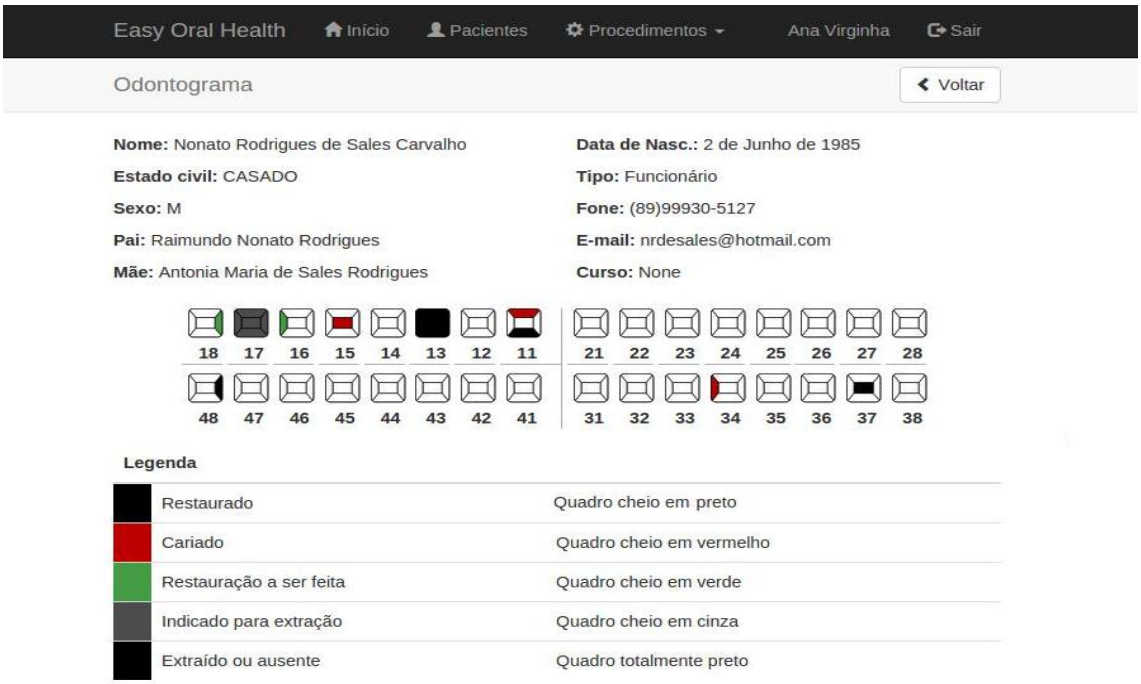

Figura 14: Odontograma

4 CONSIDERAÇÃOES FINAIS

Esse projeto foi desenvolvido com a finalidade de apresentar um sistema para auxiliar no setor de odontologia da Universidade Federal do Piauí – Campus Senador Helvídio Nunes de Barros. A implementação do sistema foi proposta para a plataforma *web* utilizando técnicas de responsividade presentes em algumas das ferramentas utilizadas, permitindo aos usuários utilizar o sistema em qualquer dispositivo que tenha acesso a *internet,* com telas adaptáveis e uma navegabilidade agradável.

Foram demonstrados os passos para o desenvolvimento do sistema, começando pelo estudo do problema, verificando as atividades relacionadas ao Serviço Odontológico. Em seguida, foram mostrados os métodos e tecnologias utilizadas para o desenvolvimento e, por último, foram mostradas as principais funcionalidades do sistema.

Atualmente o sistema encontra-se como um protótipo completamente funcional, pelo qual foram exemplificadas as funcionalidades do sistema através da captura de imagens reais que tratam das principais telas em testes simulatórios realizados.

Além dos testes simulatórios feitos em computadores do tipo *laptop,* também foram efetuados diversos testes em dispositivos móveis do tipo *smartphones,* onde houve grande destaque positivo dos recursos de responsividade, permitindo o acesso eficiente às funcionalidades em diferentes tamanhos de resolução de tela.

Como sugestões de trabalhos futuros, é possível que seja realizado um estudo de usabilidade no sistema, para tornar sua interface mais amigável, proporcionando maior facilidade de sua utilização. É necessário ainda implementar um modulo financeiro para o sistema, já que o mesmo pode ser comercializado posteriormente, além de adicionar a funcionalidade de agendamento de consultas. Também podem ser utilizadas técnicas de *real time* para mudar a forma como o servidor do sistema se comunica e troca informações com as máquinas clientes.

REFERÊNCIAS

BELEM, T. **Mas afinal, o que é o MVC?**. Disponível em: http://www.blog.thiagobelem.net/o-que-e-o-mvc/. Acesso em: 06/06/2015.

BORGES, L. E. **Python para Desenvolvedores**. 2. ed. Rio de Janeiro: Edição do Autor, 2010.

BRAUDE, E. **Projeto de Software. Da programação à arquitetura: uma abordagem baseada em Java**. Bookman, Porto Alegre, Brasil, 2005.

COCKBURN, A., **Agile Software Development**, Addison-Wesley, 2002.

CRUZ, Tadeu. **Sistemas de Informações Gerenciais**. Atlas, São Paulo, Brasil, 2009.

DOCUMENTATION, D. **Django V1.9 Documentation**. In: Django Brasil – Web Site da Comunidade Brasileira. Disponível em http://www.docs.djangobrasil.org/. Acesso em 20 de nov. 2015.

DOCUMENTATION. **MySql Documentation**. In: MySql – Web Site Oficial. Disponível em http://www.dev.mysql.com/doc/. Acesso em 07 de jan. 2016.

EIS, Diego; FERREIRA, Elcio. **HTML5 e CSS3: com farinha e pimenta**. Tableless, São Paulo, Brasil, 2012.

FREEMAN, E. **Head First HTML with CSS e XHTML**. 2007. Estilos em páginas HTML.

GORDON, Steven R.; GORDON, Judith R. **Sistemas de Informação: Uma Abordagem Gerencial**. GEN, Rio de Janeiro, Brasil, 2006.

GUEDES, G. T. A. **UML 2: Uma Abordagem Prática**. Novatec, São Paulo, Brasil, 2011.

HEUSER, Carlos Alberto. **Projeto de Banco de Dados**. Bookman, Porto Alegre, Brasil, 2009.

JUNIOR, L.A.Z.; VIDAL, A.G.R. **Construção de sistemas de informação baseados na Tecnologia Web**. *Revista de Administração da Universidade de São Paulo*, São Paulo, v.41, p.232-244, jul./ago./set. 2006.

LASTRES, H. M. M; FERRAZ, J. C. **Economia da Informação, do Conhecimento e do Aprendizado**. In Lastres, H.M.M. e Albagli, S. (coords) Informação e Globalização na Era do Conhecimento (Campus, Rio de Janeiro, 1999)

MAGNO, Alexandre. **Mobile First Bootstrap** Develop advanced websites optimized for mobile devices using the Mobile First feature of Bootstrap**.** 1 ed. Birminghan: Packt Publishing, 2013.

MATTSSON. M. (2000): **Evolution and Composition Object-Oriented Frameworks**, PhD Thesis, University of Karlskrona/Ronneby, Department of Software Engineering and Computer Science.

MILANI, A. **MySQL** : Guia do Programador. São Paulo: Novatec. 2007. 400 p.

MULBERT, Ana Luísa; AYRES, Nilce Miranda. **Fundamentos para Sistemas de Informação**. Palhoça UnisulVirtual, 2005.

PÁDUA, WILSON DE. **Engenharia de Software: fundamentos, métodos e padrões**. Editora LTC, 2000.

PADUA, W. **Engenharia de Software: Fundamentos, Métodos e Padrões**. 2. ed. Rio de Janeiro: Editora LTC, 2003. v. 1. 604 p.

PRESSMAN, R.S. **Engenharia de Software**. 6ª Ed, McGraw-Hill, 2006.

REIS, C. R. **Python na Prática: Um curso objetivo de programação na linguagem Python**. 2002. Disponível em: <http://old.kov.eti.br/programacao/python/material-curso/www.async.com.br/projects/python/pnp/>. Acesso em: 28 maio 2015.

RUBIRA, I. R. F.; RODRIGUES, C. B. F. **Odontograma e Notação Dental:** Considerações gerais. Rev. Odontol. USP. São Paulo, n.2, v.2, p.104-108, abr/jun.1988.

SABBAGH, RAFAEL. **SCRUM GESTÃO ÁGIL PARA PROJETOS DE SUCESSO**. CASO DO CÓDIGO. SÃO PAULO, 2013

SANTANA, O. e GALESI, T. **Python e Django**. São Paulo: Novatec, 2010.

SANTOS, Elisabete da Silva. **Apostila JavaScript.** FATEC, São Paulo, Brasil, 2009.

SOUZA, C. B. **Autoria de Artefatos de Software**. Dissertação de Mestrado, Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, Brasil (2008).

SCHWABER E SUTHERLAND, KEN; JEFF; **Um guia definitivo para o Scrum: as regras do jogo**. 2011. Disponível em <http://www.scrum.org/Portals/0/Documents/Scrum%20Guides/Scrum%20Guide%20-%20Portuguese%20BR.pdf> Acessado em 25/08/2015.

SPURLOCK, Jake. **Responsive Web Development Bootstrap.** 1 ed. Sebastopol: O'Reilly Media, 2013.

STARK, Jonathan; JEPSON, Brian. **Construindo Aplicativos Android com HTML, CSS e JavaScript**. Novatec, São Paulo, Brasil, 2012.

TERMO DE AUTORIZAÇÃO PARA PUBLICAÇÃO DIGITAL NA BIBLIOTECA "JOSÉ ALBANO DE MACEDO"

Identificação do Tipo de Documento

( ) Tese

( ) Dissertação

(x) Monografia

( ) Artigo

Eu, **Micael de Araujo Bastos Costa**, autorizo com base na Lei Federal nº 9.610 de 19 de Fevereiro de 1998 e na Lei nº 10.973 de 02 de dezembro de 2004, a biblioteca da Universidade Federal do Piauí a divulgar, gratuitamente, sem ressarcimento de direitos autorais, o texto integral da publicação **Ferramenta computacional de auxílio ao serviço Odontológico – SEOD - UFPI** de minha autoria, em formato PDF, para fins de leitura e/ou impressão, pela internet a título de divulgação da produção científica gerada pela Universidade.

Picos-PI 09 de Março de 2016.

Micael de Araujo Bastos Costa

Assinatura