SISTEMA DE GESTÃO PARA CLÍNICAS ODONTOLÓGICAS

MANAGEMENT SOFTWARE FOR DENTAL CLINICS

Matheus G. Zanetoni[1], Lígia R. Prete[2]

[1]Faculdade de Tecnologia Prof. José Camargo – Fatec Jales, matheus.zani@fatec.sp.gov.br
[2]Faculdade de Tecnologia Prof. José Camargo – Fatec Jales, ligia.prete@fatec.sp.gov.br

Informação e Comunicação
Subárea: Banco de Dados, Engenharia e Desenvolvimento de Software

RESUMO

A tecnologia da informação, por meio de sistemas computacionais, transformou o modo de como as organizações são geridas atualmente. Sua importância no setor de serviços é relevante devido ao elevado processamento de dados na tomada de decisão das empresas. Especificamente na área da odontologia, estes sistemas são notórios, já que possibilitam um controle de gestão eficiente para clínicas, tais como gestão financeira, controle de pacientes, agendamento de consultas, entre outros. Sendo assim, o objetivo deste trabalho consiste no desenvolvimento de um sistema *web* destinado às clínicas de odontologia que almejam tal ferramenta para auxiliar em seu processo de gestão. O sistema possui quatro módulos de operações, sendo a área do dentista, funcionários, pacientes e administrador. Ele apresenta uma interface clara e concisa que possibilita a facilidade de seu uso, além de funcionalidades essenciais de administração de clínicas, em destaque a emissão de relatórios e geração de gráficos. A modelagem do sistema foi embasada em diagramas da UML. Em relação à linguagem de programação empregou-se *Java Server Pages* em conjunto com as tecnologias HTML, CSS, JavaScript e o banco de dados PostgreSQL. O sistema intitulado OdontoSys foi implantado para testes na clínica da cirurgiã dentista Bruna Campoli, em que foram baseados os requisitos do *software*. Por fim, foi possível concluir que o sistema atende integralmente as demandas solicitadas pela clínica e que poderá ser aprimorado conforme o uso por outros profissionais de mesma área.
Palavras-chave: sistema *web*; OdontoSys; clínica; odontologia.

ABSTRACT

The information technology, through computer systems, has transformed the way organizations are managed nowadays. Its importance in the services sector is relevant due to the elevated data processing in the decision-making process in organizations. Specifically in the odontology field, these systems are notorious, once they enable an efficient management control for clinics, such as finance management, control of patients, appointments, among others. This way, the objective of this academic paper consists of the development of a web software destinated to odontology clinics which long for such tool to help in their management process. The software has four operation modules, which are the dentist's, employees', patients', and administrator's areas. It presents a clear and concise interface that allows the facility of its use, besides essential clinics' administration functionalities, highlighting report emission and graphic generation. The modeling of the software was based on the UML diagrams. Regarding the programming language, it was applied Java Server Pages along with HTML, CSS, and JavaScript technologies and the database PostgreSQL. The software entitled OdontoSys has been implanted for tests at surgeon dentist Bruna Campoli's clinic, where the software requisites were based on. In the end, it was possible to conclude that the software fully fulfills all demands solicited by the clinic and will be able to be improved as it's used by other professionals of the same field.
Keywords: web software; OdontoSys*; clinic; odontology.*

1 INTRODUÇÃO

Atualmente os sistemas de informação são relevantes ao serem utilizados na administração de um negócio, haja vista que um programa de computador é capaz de ser alimentado com dados, em seguida processá-los, de forma que o resultado seja uma informação imprescindível à tomada de decisão, a qual é crucial para a continuidade e prosperidade do negócio, a fim de melhorar os produtos e/ou serviços oferecidos pela organização. (NASCIMENTO, 2018; LAUDON; LAUDON, 2004).

Sendo assim, este trabalho justifica-se pela análise de como é gerenciada a clínica odontológica da cirurgiã dentista Bruna Campoli, situada na cidade de Jales/SP, em que foi verificada a inexistência de uma ferramenta digital que a auxilie na organização dos dados da clínica.

Embasado na justificativa do parágrafo anterior e, uma vez requerido o interesse pela aquisição do sistema por parte da cirurgiã dentista, este trabalho possui como principal objetivo expor todo o processo de levantamento de requisitos, modelagem e desenvolvimento de um sistema *web* para a gestão eficaz de clínicas odontológicas. Assim, definiu-se o nome intitulado como OdontoSys e o logotipo ilustrado na **Figura 1**.

Figura 1 – Nome e logotipo do sistema *web* para clínicas odontológicas

Fonte: Elaborado pelos autores.

Nas próximas seções serão apresentados o referencial teórico, bem como os métodos e ferramentas utilizados para o desenvolvimento do sistema, além de uma análise/discussão dos resultados obtidos e as considerações finais de encerramento deste trabalho.

2 REFERENCIAL TEÓRICO

Nesta seção, será apresentado o que há na área de pesquisa, similar a este trabalho, apresentando alguns *softwares* de gestão de clínicas odontológicas, os quais foram desenvolvidos por outros autores em trabalhos acadêmicos.

Andrade (2009) desenvolveu um sistema integrado de gestão de clínica odontológica intitulado Odonto. De acordo com o próprio autor, seu principal objetivo era permitir a gestão da informação interna da clínica, além de informação sigilosa sobre cada paciente cadastrado no sistema e seu histórico. Outros recursos relevantes neste sistema é a gestão financeira e faturamento das consultas dos pacientes.

Odonto foi desenvolvido para ambiente *web* com a utilização do banco de dados Oracle e da plataforma *Oracle Application Express*. Essa plataforma é uma ferramenta de desenvolvimento em que pode ser usado a mesma técnica para criar formulários, relatórios e gráficos, bem como a integração desses componentes em conjunto com métodos de navegação. O ambiente proporciona aos utilizadores criar praticamente qualquer aplicação por meio de um processo fácil de desenvolvimento declarativo (ANDRADE, 2009).

Dutra e Souza (2016) realizaram o desenvolvimento de outro sistema cogitado para a administração interna de clínicas de odontologia intitulado SisOdonto. Segundo os autores, este

software tem como objetivo solucionar alguns problemas, tais como o atraso nos atendimentos da Clínica Sorridente, localizada em Manhuaçu/MG, no distrito de Vilanova, tratando-se, portanto, de um caso real. O *software* foi desenvolvido para ambiente *desktop* com a utilização da linguagem de programação C# e o banco de dados MySQL.

Por fim, Ferreira, Alencar e Silva (2019), propuseram a criação de um sistema ERP para administrar uma clínica odontológica intitulado Odonto AJRT. De acordo com os autores, a solução visa facilitar o dia a dia do profissional da área de odontologia, a fim de proporcionar uma maior eficiência na gestão interna da clínica.

Odonto AJRT foi desenvolvido em formato desktop, com a utilização da linguagem de programação C# e do banco de dados MySQL, assim como foi proposto por Dutra e Souza (2016). Além disso, este sistema utilizou metodologias semelhantes às adotadas neste trabalho, tanto relacionadas às ferramentas de desenvolvimento, quanto aos métodos norteadores, tais como levantamento de requisitos e o modelo de desenvolvimento incremental.

Em suma, nesta seção foi apresentada uma pesquisa bibliográfica referente aos trabalhos similares da área de pesquisa deste estudo. Eles serão comparados com a solução desenvolvida neste artigo, na seção de análise e discussão dos resultados, a fim de verificar se a proposta deste trabalho apresenta funcionalidades ou módulos inovadores.

3 METODOLOGIA

Nesta seção serão apresentados os processos de levantamento de requisitos, análise, modelagem e desenvolvimento do sistema.

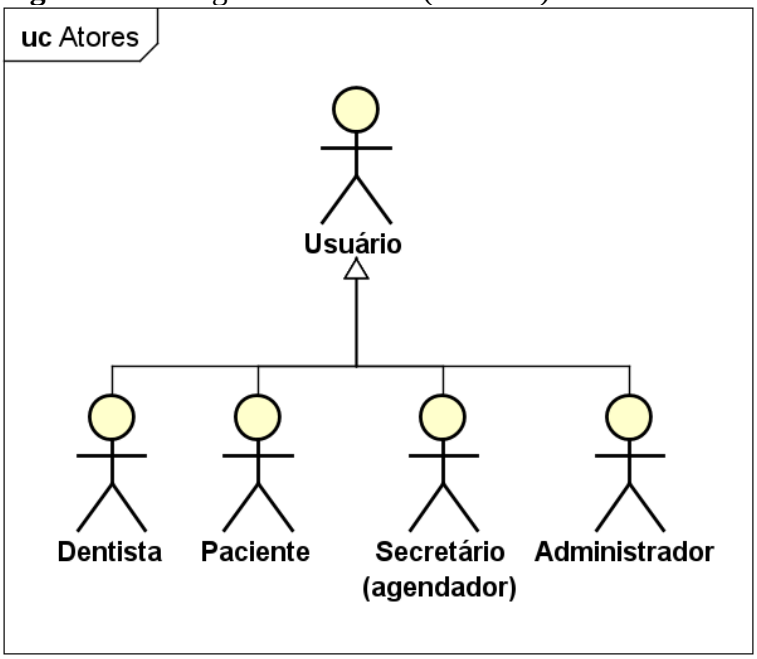
Realizada a fase de levantamento de requisitos com o cliente, a qual Pressman (2011) define como o momento de especificação do problema, sua solução e o que fará parte dela para resolvê-la (requisitos), constatou-se a necessidade de o *software* possuir quatro modularizações, sendo dentista, secretário (ou agendador), paciente e administrador.

Devido à primordialidade de quatro módulos e a fim de nortear os ciclos de desenvolvimento do sistema, foi aplicado o método de desenvolvimento incremental, que consiste na entrega do produto em fases, com aprovação do cliente e, que serve de base para o desenvolvimento das próximas etapas da solução (PRESSMAN, 2011).

Em relação à modelagem, foram elaborados diagramas embasados na UML (do inglês, *Unified Modeling Language* - Linguagem de Modelagem Unificada), em que Guedes (2011) afirma ser uma auxiliar dos engenheiros no processo de modelagem de *software*. A ferramenta Astah UML possibilitou a confecção dos diagramas de modo coerente.

Dentre os diagramas da UML, foram usados neste trabalho o diagrama de atores e de classes. O diagrama de atores apresenta o papel dos usuários na utilização das funcionalidades do sistema (GUEDES, 2011). Por outro lado, o diagrama de classes fornece uma visão inerte do *software*, ou seja, como seus componentes (classes) estão organizados e interrelacionados (PRESSMAN, 2011). As **Figuras 2 e 3** ilustram os diagramas confeccionados.

Figura 2 — Diagrama de atores (usuários)



Fonte: Elaborado pelos autores.

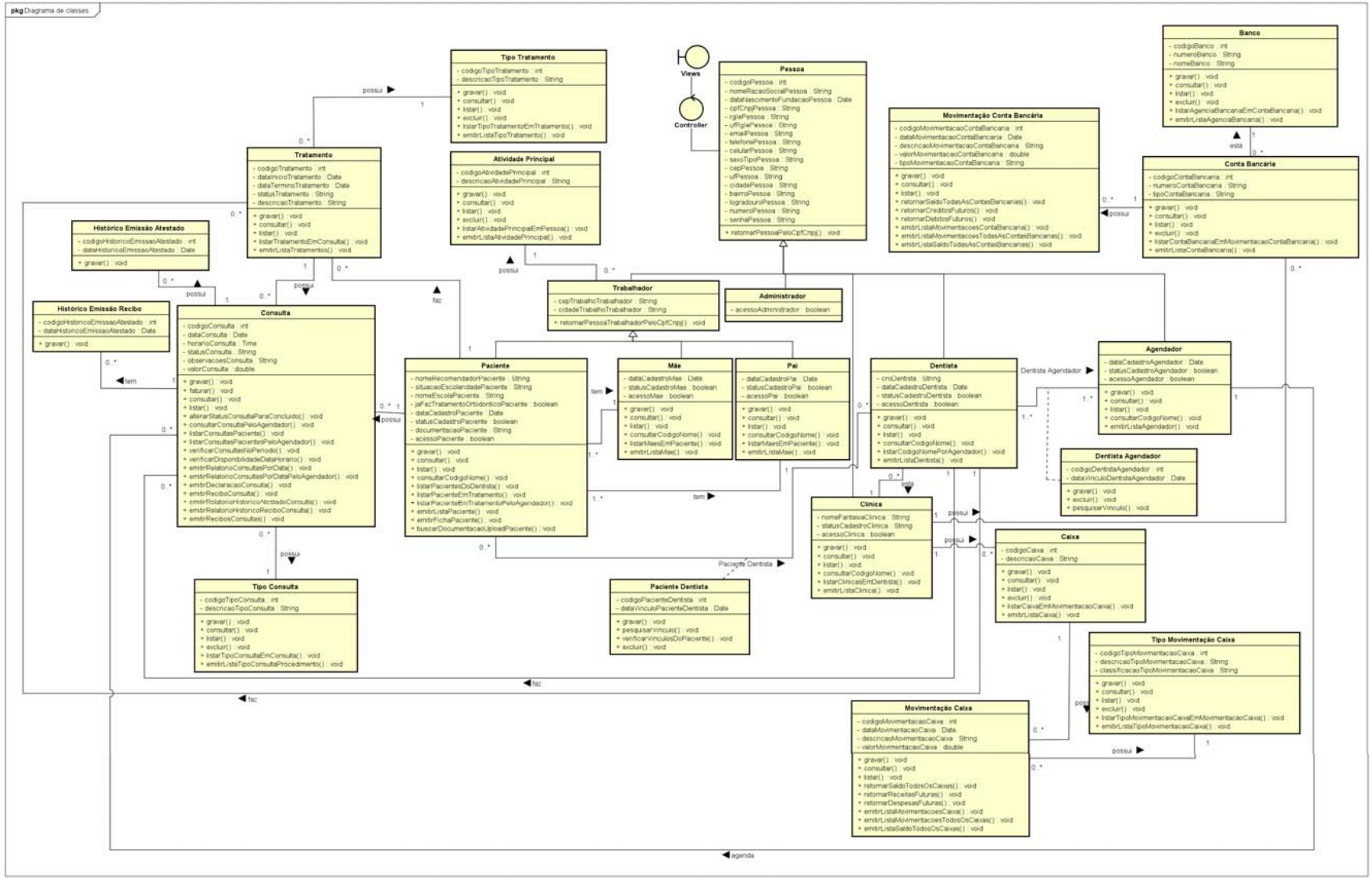
Embasado no Diagrama de Atores, o usuário **Dentista** é o responsável pela clínica. Dentre suas funções, as principais são o cadastro de pacientes/tratamentos, agendamento de consultas e movimentações do caixa/conta bancária da clínica.

O usuário **Paciente** é aquele consultado pelo dentista. Seu acesso ao sistema limita-se apenas em verificar se há ou não consultas agendadas pelo dentista ou secretário.

O usuário **Secretário** (ou agendador) é o responsável por agendar as consultas do dentista. Ele pode, por exemplo, efetuar o cadastro de um novo paciente que começará a frequentar a clínica, porém sua função principal resume-se em agendamentos.

Por fim, o usuário **Administrador** é o responsável por cadastrar dentistas, clínicas e secretários, bem como conceder ou revogar o acesso deles nos módulos do sistema, além de alterar configurações globais. Vale ressaltar que no caso de uma clínica odontológica autônoma, o dentista acaba assumindo o papel de administrador.

Figura 3 — Diagrama de classes



Fonte: Elaborado pelos autores.

A arquitetura do *software* está baseada no modelo MVC (do inglês *Model/View/Controller* – Modelo/Visão/Controlador) que é um padrão de arquitetura de *software* e padroniza a segregação em três camadas. A camada Visão (*View*) é a visualização dos dados, ou seja, as páginas *web* (interface gráfica com o usuário). A camada Modelo (*Model*) contém a essência da regra de negócio do cliente, isto é, as classes envolvidas e suas relações lógicas. Por fim, a camada Controlador (*Controller*) atua como um intermediário entre a camada de Visão e as demais camadas gerenciando o fluxo de dados (PRESSMAN, 2011).

Ao modelo MVC foi adicionada a camada DAO (do inglês, *Data Access Object* – Objeto de Acesso a Dados) que é responsável pela persistência dos dados, ou seja, pela comunicação e realização de operações no banco de dados. Essa nova camada surgiu devido à necessidade de se separar os métodos de manipulação de dados da camada *Model*, tornando tais processos completamente transparentes, ou seja, encapsulados em relação às demais camadas do *software* (MEDEIROS, 2016).

A linguagem de programação em que o *software* se baseia é Java usando a tecnologia JSP (*Java Server Pages*) que auxilia os desenvolvedores a criarem páginas *web* geradas dinamicamente baseadas em HTML, XML ou outros tipos de documentos, além de ser orientada a objetos. Kay ([1966?] apud GUANABARA, 2016), considerado o pai da orientação a objetos, postulou que:

> O computador ideal deve funcionar como um organismo vivo, isto é, cada célula se relaciona com outras a fim de alcançar um objetivo, mas cada uma funciona de forma autônoma. As células poderiam também reagrupar-se para resolver um outro problema ou desempenhar outras funções.

Trazendo para uma outra perspectiva o que foi dito por Kay ([1966?] apud GUANABARA, 2016), as células mencionadas são as classes, as quais são tipos abstratos de dados que instanciam, ou seja, criam objetos abstratos a partir delas. Esses objetos são abstrações computacionais de objetos do mudo real e são compostos por características (atributos) e métodos (ações). Em um programa de computador orientado a objetos, estes podem executar procedimentos de maneira isolada ou podem se relacionar com outros objetos para executar procedimentos que exijam essa interrelação, a fim de o programa cumprir sua tarefa.

A base de dados do *software* respalda no modelo relacional. Este modelo, segundo Amadeu (2014, p. 61) “[...] representa o banco de dados como uma coleção de relações. Informalmente, cada relação é semelhante a uma tabela de valores (composta por linhas e colunas) [...]”.

A interação entre o banco de dados e a linguagem de programação Java está baseada no Modelo Objeto Relacional. Essa técnica possui como principal objetivo traduzir instruções orientadas a objetos para instruções orientadas ao modelo relacional do banco de dados por meio da criação da camada de persistência de dados (DAO) (BERNARDI, 2008).

Os aspectos da interface do *software*, tais como *design* e interações nas páginas *web*, estão fundamentadas principalmente em HTML 5, CSS 3 e JavaScript. Essas tecnologias simplificam o entendimento do código-fonte, suportam novos conteúdos, como vídeos e padronizam o desenvolvimento de páginas *web* para os diversos dispositivos que possuem acesso à Internet (TAVARES, 2012).

Assim sendo, por meio das tecnologias mencionadas foram construídas interfaces *web* embasadas principalmente nos conceitos de usabilidade e interatividade entre o usuário e o *software*. Isso se faz necessário, visto que em um sistema de informação, tanto a interação com o usuário, quanto a visualização das interfaces, precisam ter qualidade para melhor aproveitamento dos recursos computacionais oferecidos pelo *software* (BARBOSA; SILVA, 2010).

As interfaces *web* do *software* e suas interações, do mesmo modo, estão baseadas nos *frameworks*[1] Bootstrap e jQuery. O Bootstrap contém bibliotecas de estilo prontas que auxiliam o desenvolvimento *front-end*[2] (BOOTSTRAP..., 2021). Já o jQuery é uma biblioteca JavaScript desenvolvida e pensada para reduzir o trabalho de programação de *scripts* que são interpretados pelo navegador do usuário. Com ela é possível, por exemplo, criar requisições assíncronas para outras camadas do *software*, sem a necessidade de recarregar a página *web* (TEIXEIRA, 2013).

O versionamento do *software* foi realizado e embasado no conceito de repositório distribuído com a utilização dos sistemas de controle de versão distribuídos Git e GitHub. De acordo com Straub e Chacon (2021, p. 10, tradução nossa):

> [...] Em Sistemas de Controle de Versão Distribuídos [...] os colaboradores não somente veem a última versão dos arquivos; por outro lado, eles completamente clonam o repositório, incluindo seu histórico completo. Então, se algum servidor falhar, e esses colaboradores estavam colaborando através daquele servidor, qualquer repositório desses colaboradores pode ser copiado de volta para o servidor para ser restaurado. Cada clonagem é realmente um backup de todos os dados.

Em outros termos, e complementando, em um sistema de controle de versão de código-fonte distribuído, cada desenvolvedor possui todo o histórico de versões do *software* localmente, obtido através de um repositório hospedado em um servidor, o qual pode ser gerido pela ferramenta GitHub, por meio do processo denominado clonagem. Por ter todo o histórico, cada desenvolvedor consegue voltar e avançar na linha do tempo de desenvolvimento, bem como fazer modificações, confirmá-las e enviá-las ao repositório do servidor utilizando, por exemplo, a ferramenta Git. No caso de uma falha no servidor central, é possível fazer o envio do repositório local do desenvolvedor novamente para o servidor. E no caso de uma falha no instrumento de trabalho do desenvolvedor, é possível clonar novamente o repositório do servidor central em outro dispositivo de trabalho, trazendo todo o histórico do projeto.

O Diagrama de Entidade-Relacionamento do banco de dados do *software* (modelo lógico) foi modelado por meio da ferramenta CASE (do inglês, *Computer-Aided Software Engineering*) SQL Power Architect que agiliza todo o processo fazendo com que não seja necessário utilizar código SQL, visto que a ferramenta, ao final, gera todo o código (modelo físico) com base no que foi modelado.

A ferramenta pgAdmin 4 foi empregada para gerenciar o Banco de Dados PostgreSQL. Ela é responsável pela execução do código SQL gerado pela aplicação SQL Power Architect e pela gestão do banco de dados do *software*.

O Ambiente de Desenvolvimento Integrado utilizado durante o processo de desenvolvimento foi o NetBeans IDE 12.5, que permite programar a lógica da regra de negócio em JSP, utilizando-se a arquitetura MVC e as páginas *web* com HTML 5, CSS 3, JavaScript e *frameworks* auxiliares de desenvolvimento (Bootstrap e jQuery) de forma integrada, além de executar a aplicação por meio de um servidor *web* para testes e validações, sendo o utilizado o Apache Tomcat 9.

Alguns ícones e imagens que compõem o *software* foram extraídos da Internet em sites gratuitos, como o IconFinder, porém outros foram confeccionados com a utilização dos *softwares* Adobe Photoshop 2021 e Inkscape 1.1.1, que são aplicações destinadas para desenho, criação e edição de imagens.

---

[1] Um *framework* funciona como uma espécie de biblioteca de arquivos que armazena diversas funções prontas para o desenvolvimento de uma aplicação *web*, reduzindo a complexidade que há ao se desenvolver uma aplicação *web* sem qualquer recurso.

[2] O termo *front-end* diz respeito ao desenvolvimento das interfaces *web* gráficas de um *software*, isto é, as páginas *web* com as quais o usuário interagirá diretamente.

Por fim, a aplicação utilizada para a confecção dos relatórios emitidos pelo *software* foi o Jaspersoft Studio, desenvolvido em Java e, possuindo assim, total integração com *softwares* também desenvolvidos nesta linguagem de programação.

Os arquivos dos relatórios são criados com a extensão JRXML e são tratados por meio de funções da biblioteca Jasper, fazendo com que o *software* busque os dados pertinentes no banco de dados, carregue o arquivo, inclua esses dados e gere finalmente um arquivo PDF que é aberto no próprio navegador do usuário, com a opção de imprimi-lo ou salvá-lo.

4 ANÁLISE E DISCUSSÃO DOS RESULTADOS

A partir dos métodos, técnicas e ferramentas utilizados com base no referencial teórico apresentado para o desenvolvimento do *software*, constatou-se que os resultados foram satisfatórios. Os requisitos do cliente foram atendidos minuciosamente a fim de haver simplicidade na realização das operações por parte do *software*, visto que os possíveis usuários que poderão utilizá-lo, além da Dra. Bruna Campoli, podem ser profissionais de odontologia não muito adeptos com a tecnologia.

De acordo com a Dra. Bruna, o *software* enquadra-se com o solicitado na fase de levantamento de requisitos. Mediante os testes realizados, nos vários ciclos de desenvolvimento, o sistema foi aprovado com todos os módulos e funcionalidades disponibilizados.

As **Figuras de 4 a 10** mostram algumas interfaces e funcionalidades principais do sistema como resultado do trabalho.

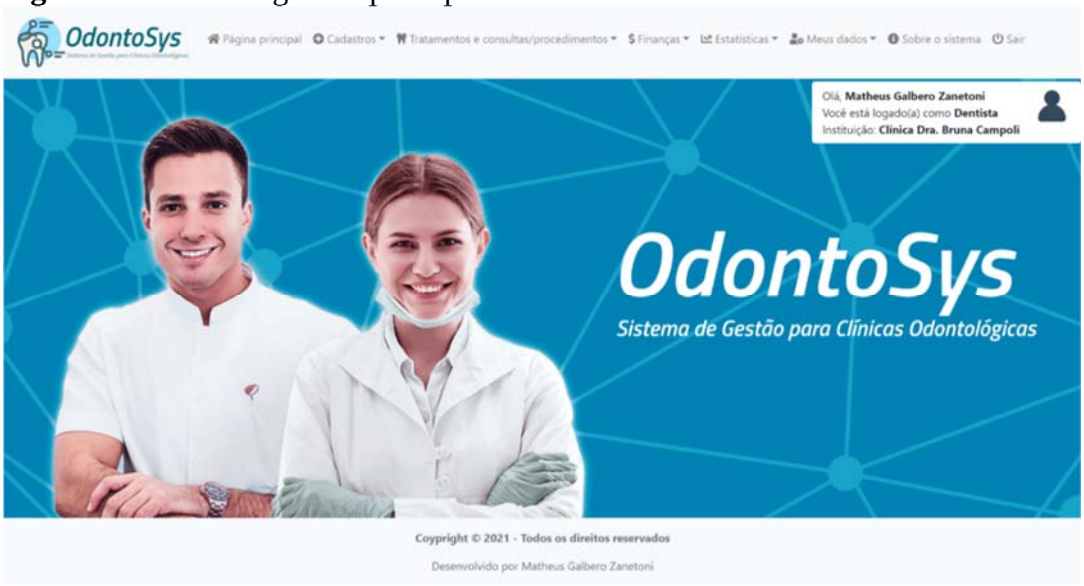
Na **Figura 4** é ilustrado a tela de login do sistema, em que o usuário informará seu CPF, selecionará a clínica que está associado, o módulo que deseja acessar e ao final informará a senha para acessá-lo.

Figura 4 – Interface gráfica de login do sistema

Fonte: Elaborado pelos autores.

A **Figura 5** apresenta a tela principal do módulo do dentista. Ele pode acessar as funcionalidades disponíveis para o seu usuário no menu superior. Para cada módulo do sistema, o menu é carregado de maneira diferente.

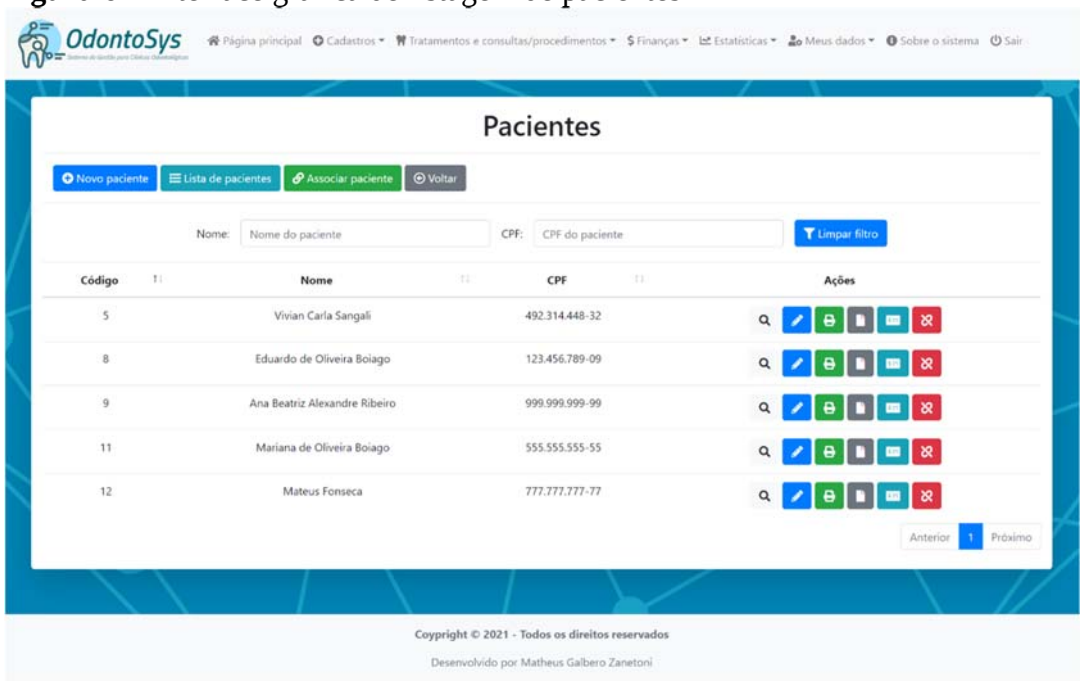
Figura 5 – Interface gráfica principal do módulo do dentista



Fonte: Elaborado pelos autores.

A **Figura 6** exibe a tela de listagem de pacientes. A partir dela, o dentista pode cadastrar, vincular ou desvincular um paciente, bem como emitir uma lista dos pacientes já cadastrados, consultar e editar seus dados, além de emitir alguns documentos, tais como a ficha cadastral, e recibos de consultas já realizadas.

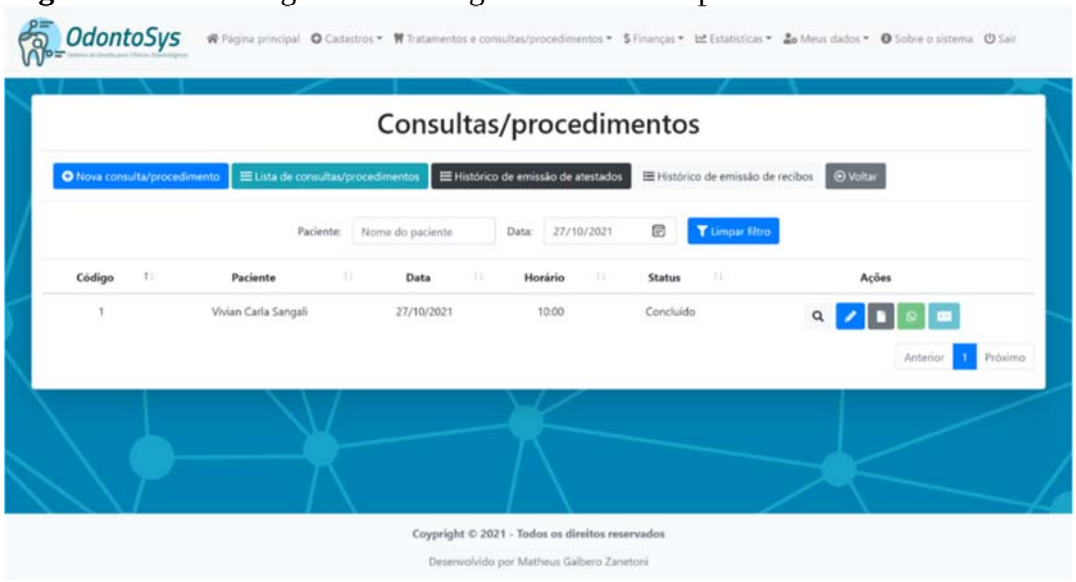
Figura 6 – Interface gráfica de listagem de pacientes



Fonte: Elaborado pelos autores.

A **Figura 7** expõe a tela de consultas/procedimentos agendados para os pacientes. A partir dela, é possível agendar uma nova consulta/procedimento, contatar o paciente via WhatsApp, emitir a lista de consultas em um período, emitir os históricos de emissões de recibos e atestados, bem como consultar e editar os dados da consulta, além de gerar o atestado e o recibo.

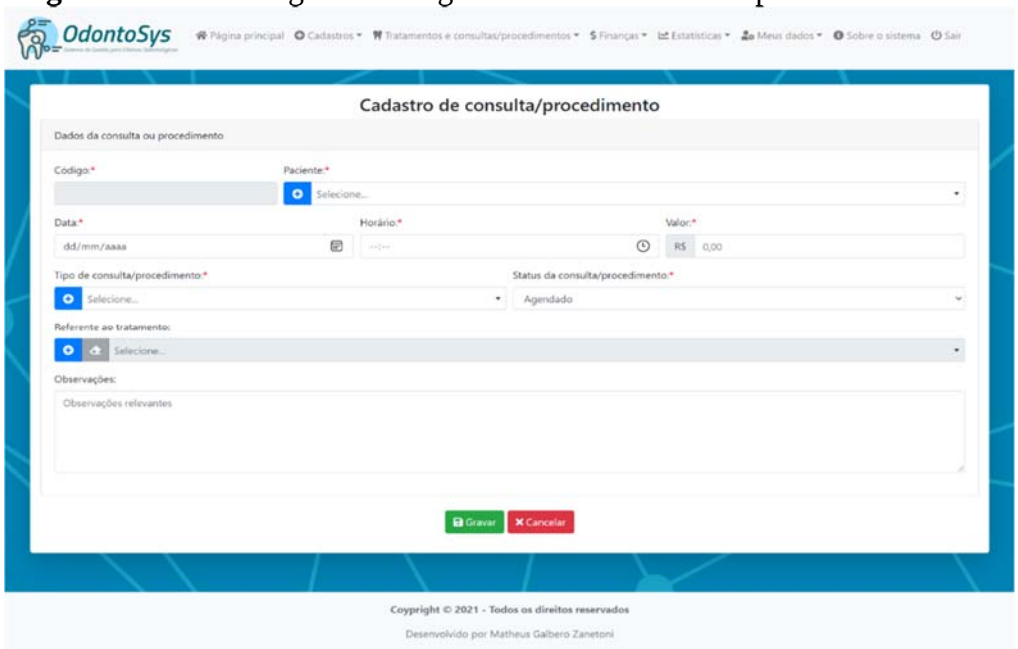
Figura 7 – Interface gráfica de listagem de consultas/procedimentos



Fonte: Elaborado pelos autores.

A **Figura 8** ilustra a tela de cadastro de consulta/procedimento. O dentista deverá informar os dados requisitados e, ao final, pressionar o botão de gravação para agendar. Ele ainda pode cancelar o agendamento nesta tela a qualquer momento.

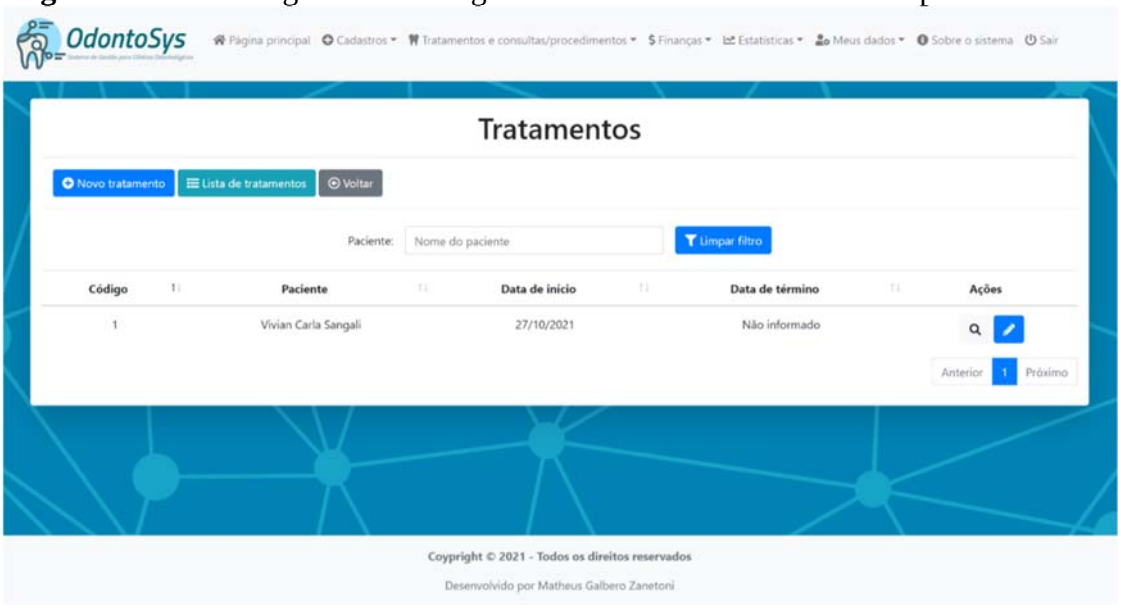
Figura 8 – Interface gráfica de agendamento de consulta/procedimento



Fonte: Elaborado pelos autores.

A **Figura 9** mostra a tela de listagem de tratamentos que um ou mais pacientes estão recebendo. A partir dela, o dentista pode cadastrar um novo tratamento para um paciente, emitir a lista de pacientes e seus respectivos tratamentos, bem como consultar e editar os dados destes tratamentos concedidos.

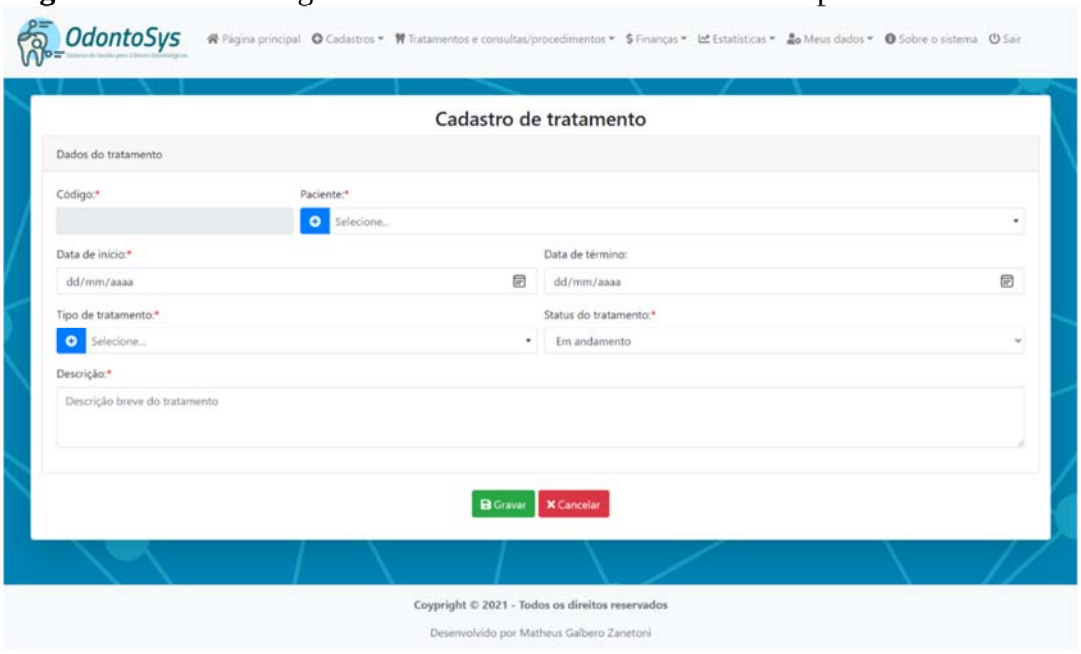
Figura 9 – Interface gráfica de listagem de tratamentos concedidos ao paciente



Fonte: Elaborado pelos autores.

A **Figura 10** exibe a tela de cadastro de tratamentos para os pacientes. O dentista deverá informar os dados requisitados e, ao final, pressionar o botão de gravação para cadastrar o tratamento. Ele pode cancelar o cadastro nesta tela a qualquer momento.

Figura 10 – Interface gráfica de cadastro de tratamento de um paciente



Fonte: Elaborado pelos autores.

A **Figura 11** apresenta a fase de testes do sistema realizado pela cliente, a cirurgiã dentista Bruna Campoli.

Figura 11 – Dra. Bruna Campoli testando o *software*

Fonte: Elaborado pelos autores.

Comparando OdontoSys aos *softwares* citados no referencial teórico deste trabalho, concluiu-se que todos possuem as mesmas funcionalidades básicas, tais como cadastros, agendamentos de consultas, emissões de relatórios e geração de gráficos. Contudo, OdontoSys não contempla todas as funcionalidades dos *softwares* mencionados anteriormente, visto que foi elaborado para atender as necessidades básicas da clínica da Dra. Bruna Campoli, que não possui todos os requisitos encontrados nas soluções expostas. Porém, diversamente de SisOdonto e Odonto AJRT, OdontoSys é um sistema *web*, sendo capaz de ser acessado de qualquer lugar. Do mesmo modo, poderá ser aprimorado, a fim de atender outras necessidades que possam surgir com o decorrer do tempo, tanto por parte desta cliente, quanto por parte de outros profissionais da área de odontologia e do desenvolvedor responsável pelo trabalho.

5 CONSIDERAÇÕES FINAIS

Diante do trabalho realizado e com o *software* na fase final de testes, concluiu-se que as atividades de gestão da clínica da Dra. Bruna Campoli se tornaram satisfatórias. Uma vez que a cliente utilizará em breve o sistema para gerenciar a clínica, e logo, fará a coleta de dados dos seus pacientes para armazenar, processar e gerar informações por meio de consultas e relatórios.

A previsão de implantação e funcionamento definitivo do *software* na clínica da cliente supracitada está previsto para ocorrer em janeiro de 2022. No momento, ela dispõe de uma versão de testes candidata à versão final. Até a implantação definitiva, mais testes serão realizados para averiguar a necessidade de reparos no código-fonte, caso algum erro ocorra ou um novo requisito seja identificado, tanto pela cliente, quanto pelo desenvolvedor.

Vale reforçar que este *software* é customizável e poderá ser implantado em outras clínicas de odontologia que almejam uma solução para serem gerenciadas. Possíveis ajustes poderão ser realizados, devido as diversidades nas regras de negócios de cada clínica, e sendo assim, melhorias com o surgimento de novos requisitos. Por fim, o *software* atenderá todas as demandas e cumprirá satisfatoriamente suas tarefas.

REFERÊNCIAS

AMADEU, C. V. (org.). **Banco de dados**. São Paulo: Pearson Education do Brasil, 2014. (Coleção bibliográfica universitária Pearson). *E-book*.

ANDRADE, E. C. B. **Desenvolvimento de sistema integrado de gestão de clínica odontológica – Odonto**: módulo de registro de dados. 2009. Trabalho de Conclusão de Curso (Licenciatura em Tecnologias de Informação e Comunicação) – Universidade de Cabo Verde, Palmarejo, 2009.

BARBOSA, S. D. J.; SILVA, B. S. **Interação humano-computador**. Rio de Janeiro: Elsevier, 2010.

BERNARDI, D. A. **Técnicas de mapeamento objeto relacional Revista SQL Magazine 40**. 2008. Disponível em: https://www.devmedia.com.br/tecnicas-de-mapeamento-objeto-relacional-revista-sql-magazine-40/6980. Acesso em: 9 out. 2021.

BOOTSTRAP. 2021. Disponível em: https://www.devmedia.com.br/guia/bootstrap/38150. Acesso em: 9 out. 2021.

CHACON, S.; STRAUB, B. **Pro Git**: everything you need to know about Git. 2. ed. Nova York: Apress, 2021 (*E-book*).

DUTRA, R. A. F.; SOUZA, E. F. SisOdonto: sistema para gerenciamento e agendamento odontológico. **Repositório de Trabalhos de Conclusão de Curso**, Manhuaçu, v. 6, n. 1, 2016. Disponível em: http://pensaracademico.facig.edu.br/index.php/repositoriotcc/article/view/651. Acesso em: 4 nov. 2021.

FERREIRA, J. M. S.; ALENCAR, R. S.; SILVA, T. R. S. **Desenvolvimento de sistema ERP para clínica odontológica**. 2019. Trabalho de Conclusão de Curso (Graduação em Tecnologia em Análise e Desenvolvimento de Sistemas) – Faculdade de Tecnologia de Americana Ministro Ralph Biasi, Americana, 2019.

GUANABARA, G. **Curso POO teoria #01a**: o que é Programação Orientada a Objetos. [*S.l.: s.n.*], 2016. 1 vídeo (37:58 min). Publicado por Curso em Vídeo. Disponível em: https://www.youtube.com/watch?v=KlIL63MeyMY&list=PLHz_AreHm4dkqe2aR0tQK74m8SFe-aGsY. Acesso em: 10 out. 2021.

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

LAUDON, K. C.; LAUDON, J. P. **Sistemas de informação gerenciais**: administrando a empresa digital. 5. ed. São Paulo: Pearson Prentice Hall, 2004. *E-book*.

MEDEIROS, H. **Implementando o Data Access Object no Java EE**. 2016. Disponível em: https://www.devmedia.com.br/implementando-o-data-access-object-no-java-ee/33339. Acesso em: 10 out. 2021.

NASCIMENTO, A. B. (org.). **Sistemas de informação para saúde**. São Paulo: Pearson Education do Brasil, 2018. *E-book*.

PRESSMAN, R. S. **Engenharia de software**: uma abordagem profissional. 7. ed. Porto Alegre: AMGH, 2011.

TAVARES, L. A. **Primeiros passos no HTML5, CSS3 e JavaScript**. 2012. Disponível em: https://www.devmedia.com.br/primeiros-passos-no-html5-javascript-e-css3/25647. Acesso em: 9 out. 2021.

TEIXEIRA, J. R. **jQuery tutorial**. 2013. Disponível em: https://www.devmedia.com.br/jquery-tutorial/27299. Acesso em: 9 out. 2021.

AGRADECIMENTOS

Primeiramente a Deus, pela força, persistência e capacidade concedidas para a realização deste projeto.

À minha orientadora a Profª. Dra. Lígia Rodrigues Prete, pelo excelente auxílio prestado a mim nos momentos de dúvida.

À cirurgiã dentista Bruna Campoli, a qual contribuiu totalmente para que este projeto pudesse ser desenvolvido e implantado com sucesso.