

Sistema de Gestão “Pets”: Software para Clínicas Veterinárias com Integração de Comunicação.

Augusto Pinto Meine
Curso de Ciência da Computação
Universidade Franciscana
CEP 97010-032 – Santa Maria, RS, Brasil
augusto.meine@gmail.com

Sylvio André Garcia Vieira
Curso de Ciência da Computação
Universidade Franciscana
CEP 97010-032 – Santa Maria, RS, Brasil
sylvio@ufn.edu.br

Resumo—Este artigo apresenta uma aplicação *web* voltada para a gestão de estabelecimentos que atuam como instituições de clínica veterinária. O *software* possibilita o gerenciamento das atividades dos profissionais da área veterinária permitindo agilizar os processos e ter um controle fácil e intuitivo. A metodologia definida para implementação foi a *Feature-Driven Development* (FDD) focando mais no desenvolvimento e visando a separação dos planejamentos e das entregas por funcionalidade, tornando as entregas mais frequentes e com possibilidade de modificação dependendo das necessidades do usuário. A aplicação teve o desenvolvimento realizado com o uso do Spring Boot para o *back-end*, o Angular para o *front-end* e o MySQL como gerenciador de banco de dados. A aplicação desenvolvida não tem fins lucrativos, e possui foco nas instituições com menor poder aquisitivo, visando melhorar o desempenho das atividades exercidas pelos profissionais e, por consequência, permitindo um melhor atendimento clínico.

I. INTRODUÇÃO

As clínicas veterinárias são responsáveis pela manutenção da saúde e bem-estar dos animais, realizando o atendimento em consultas periódicas, aplicações de vacinas, tratamento clínico-cirúrgico, entre outras ações. Apesar da gama de ações realizadas, um das possíveis barreiras enfrentadas pelas clínicas veterinárias, que não aderiram a automatização dos processos, seria a pouca eficiência na gestão dos atendimentos e procedimentos internos.

Um dos cenários existentes ocorre nas instituições que prestam ajuda aos animais abandonados, onde existe uma alta carga de trabalho, visto que, geralmente, precisam de suporte de voluntários e doações, não conseguindo dar conta da alta quantidade de gatos e cachorros que vivem nas ruas [1], e devido a quantidade de médicos-veterinários inscritos ter aumentado de 111,2 mil em 2017 para 145,6 mil em novembro de 2020, totalizando um crescimento de 34,3 mil, está ocorrendo, em paralelo com o crescimento das clínicas, hospitais, consultórios, ambulatórios e *pet shop* em 2017 de 38,1 mil e chegando a 53,1 mil em novembro de 2020 [2], o crescimento da demanda por sistema automatizados.

Visando buscar alta eficiência na gestão do tempo torna-se interessante a implementação de um sistema de comunicação via WhatsApp para notificação, confirmação ou alteração da data da consulta marcada, reduzindo as ocorrências de ausência dos tutores e por consequência tende a reduzir o tempo ocioso dos profissionais.

Observando a demanda existente do laboratório de práticas dos cursos de computação da Universidade Franciscana (UFN), foi identificada a possibilidade de desenvolvimento de um sistema *web* que atendesse esta necessidade.

Visando auxiliar as empresas de menor porte e instituições com menor poder aquisitivo, este projeto teve como objetivo o desenvolvimento de um *software* web para gerenciar as tarefas realizadas nas clínicas veterinárias, tendo como base os seguintes objetivos específicos:

A. Objetivos específicos

- Automatizar os procedimentos manuais com uso de tecnologias apropriadas para o desenvolvimento;
- Centralizar as informações sensíveis dos clientes, dos animais e dos dados vinculados aos atendimentos;
- Realizar as comunicações e notificações via WhatsApp;

B. Justificativa

A interação dos animais com o ser humano aparece desde os tempos primitivos, os quais auxiliavam na caça, proteção do território e no transporte de cargas. Essa interação pode ter surgido a partir da relação com lobos, que desenvolveram características dóceis. O homem depende da interação com outras espécies para sua sobrevivência, antigamente considerados predadores, passando depois para domésticos [1].

A escolha das tecnologias visa a escalabilidade futura do sistema, permitindo que ele possa crescer e se adaptar às necessidades das clínicas veterinárias. A adoção de práticas e ferramentas atuais assegura que o projeto se mantenha atualizado com as tendências do setor, oferecendo uma base sólida para o desenvolvimento contínuo e eficiente.

II. REFERENCIAL TEÓRICO

Nesta seção, serão descritos conceitos essenciais para a compreensão do trabalho, além das tecnologias que foram utilizadas e que possuem amplo uso no mercado de desenvolvimento.

A. Clínica veterinária

Os animais são seres vivos que possuem algumas características em comum: nascem, desenvolvem-se, reproduzem-se e morrem. O conjunto dessas etapas é chamado de ciclo vital.

Eles podem ser classificados em vertebrados, invertebrados, selvagens, domésticos, terrestres, aquáticos e aéreos [3].

Clínicas Veterinárias são estabelecimentos destinados ao atendimento de animais para consultas e tratamentos clínico-cirúrgicos, podendo ou não ter internações, sob a responsabilidade técnica e presença de médico veterinário, onde prestam atendimento integral em uma especialidade de Medicina Veterinária [4].

B. Java

Java é uma linguagem de programação que foi disponibilizada pela primeira vez pela Sun Microsystems em 1995. Após a sua liberação teve uma grande participação no mundo digital, oferecendo confiabilidade em muitos serviços e aplicativos desenvolvidos [5], sendo considerada a linguagem de programação e plataforma de desenvolvimento mais utilizada. Com milhões de desenvolvedores executando mais de 60 bilhões de *Java Virtual Machines* (JVM) em todo o mundo, o Java continua sendo amplamente aplicada nas plataformas de desenvolvimento de empresas e desenvolvedores [6].

C. API (Application Programming Interface)

APIs são mecanismos que permite dois componentes de *software* se comunicarem entre si usando um conjunto de definições e protocolos. O termo “API” significa *Application Programming Interface* (Interface de Programação de Aplicação) e, no contexto de APIs, a “aplicação” refere-se a qualquer *software* com uma função distinta, a “*interface*” pode ser pensada como um contrato de serviço entre duas aplicações, no qual é definido como as duas se comunicam através de solicitações e respostas, sendo a aplicação que envia a solicitação chamada de “cliente” e a aplicação que envia a resposta chamada de “servidor” [7].

Uma REST API (*Representational State Transfer Application Programming Interface*) é uma *interface* que permite a comunicação entre diferentes sistemas através da *web*. Ela utiliza o protocolo HTTP (*Hypertext Transfer Protocol*) para enviar e receber dados, geralmente no formato JSON (*JavaScript Object Notation*). O Spring Boot simplifica a criação das APIs, oferecendo ferramentas e configurações que agilizam o desenvolvimento de aplicações *web*. Durante a comunicação, são empregados verbos HTTP para descrever as operações realizadas, como o *get*, usado para leitura de dados; o *post*, para salvar novas informações; o *put*, destinado à atualização de dados existentes; e o *delete*, para exclusão de dados.

D. WhatsApp

O WhatsApp Messenger é um aplicativo *mobile* que possui funcionalidades para enviar e receber mensagens, fazer chamadas e videochamadas, compartilhar fotos e vídeos, entre outras implementações vinculadas à comunicação. Surgiu como uma alternativa ao SMS, oferecendo suporte ao envio e recebimento de uma variedade de mídias como textos, fotos, vídeos, documentos, localização e chamadas de voz. Permitindo compartilhamentos pessoais e profissionais agregados a implementação

de criptografia de ponta a ponta, visando a comunicação sem barreiras [8].

A API do WhatsApp permite a integração de mensagens do WhatsApp com sistemas de negócios, possibilitando a comunicação automatizada e personalizada. A API tem foco principal as empresas, possibilitando funcionalidades como envio de mensagens, notificações e atendimento ao cliente através da plataforma do WhatsApp [9].

E. Spring Framework

O Java Spring Framework (Spring Framework) é um *framework* muito utilizado a nível empresarial, de *software* livre, para criar aplicativos independentes que são executados na *Java Virtual Machine* (JVM) [10]. O Java Spring Boot (Spring Boot) é uma ferramenta que facilita e agiliza o desenvolvimento de aplicativos da *web* e de microsserviços com o Spring Framework por meio da autoconfiguração, podendo ter configurações alteradas manualmente de forma facilitada, e possuindo a capacidade de criar aplicativos independentes. Permitindo configurar um aplicativo com poucos requisitos de instalação e configuração [10].

F. Angular

Angular é um *framework* de desenvolvimento que possui uma estrutura baseada em componentes para construir aplicações *web* escaláveis. Para auxílio é disponibilizado uma coleção de bibliotecas bem integradas que cobrem uma ampla variedade de recursos, incluindo roteamento, gerenciamento de formulários, comunicação cliente-servidor e muito mais. As ferramentas de desenvolvedor incluídas ajudam a desenvolver, construir, testar e atualizar seu código de maneira prática. Por ser escalável, o Angular possibilita dimensionar/developar projetos pequenos até aplicativos de nível empresarial, possuindo mais de 1,7 milhão de desenvolvedores, autores de bibliotecas e criadores de conteúdo [11]. A comunicação com o *back-end* é realizada através do uso de APIs, utilizando o módulo HttpClient, permitindo a troca eficiente de dados em formato JSON via protocolo HTTP, resultando em aplicações *web* completas, escaláveis e de fácil manutenção.

G. MySQL

O MySQL é amplamente conhecido no mundo, sendo um sistema de gerenciamento de banco de dados (SGBD) de código aberto, que usa a linguagem SQL como *interface*. A sequência “SQL” de “MySQL” significa “*Structured Query Language*” (linguagem de consulta estruturada), sendo a linguagem padronizada mais comum utilizada para acessar bancos de dados [12].

III. TRABALHOS CORRELATOS

Nesta seção, serão apresentados alguns artigos no qual serviram e auxiliaram como base na pesquisa do cenário atual de sistemas de gestão de clínicas veterinárias. Os trabalhos foram escolhidos devido as práticas eficazes de desenvolvimento, implementação e gestão de sistemas voltados para ambiente veterinário, em que algumas das metodologias foram

adaptadas e incorporadas ao desenvolvimento do Sistema de Gestão “Pets”, garantindo uma abordagem mais eficiente e atendendo às necessidades básicas específicas das clínicas veterinárias.

A. LocalizaPet: Desenvolvimento de um Sistema Web para Localização de Animais Perdidos, Doações para ONGs e Adoção de Animais

O projeto LocalizaPet, desenvolvido por Rafael Padilha de Meneses [1] e colaboradores, tem como foco o desenvolvimento de um sistema *web* para auxiliar a comunicação, sobre os animais de estimação, entre instituições e usuários, permitindo ao usuário comum buscar por animais cadastrados para adoção, possibilitando também a divulgação de “*pets*” perdidos e ajudar instituições necessitadas. Servindo como veículo de divulgação das instituições para juntar recursos e resolvendo questões como: consultas, remédios, cirurgias e alimentos.

A metodologia implementada foi a *Feature-Driven Development* (FDD), usando como base no desenvolvimento o Spring Boot para o *back-end*, React para o *front-end* e MongoDB para o banco de dados. A aplicação desenvolvida alcançou o objetivo notificado durante o artigo, as funcionalidades propostas foram implementadas e as páginas foram desenvolvidas.

B. VetService: Sistema De Gestão Clínica Veterinária

O projeto VetService, desenvolvido por Eliézer Leite Chaves e Gustavo Gonçalves Rezende [13], analisou que devido ao aumento de profissionais da área de clínicas veterinárias a demanda por serviços e sistemas de gestão também aumentaram. A burocracia, complexidade de uso e o aumento do preço do produto tornou menos acessível para as pequenas empresas e microempreendedores da área. Para sanar o problema, foi elaborada a ideia da criação de um sistema sem mensalidade ou cobranças.

Usando como base no desenvolvimento o PHP para o *back-end*, o Bootstrap e JavaScript para o *front-end* e MySQL para o banco de dados. A aplicação desenvolvida atendeu aos requisitos notificados no artigo, onde foi possível atender a demanda. Ficou pendente algumas implementações no sistema de Login e a adaptação para alinhamento da lei de proteção de dados. Como planejamento futuro foi comentado a possibilidade de criação de um sistema de pagamento interno para auxílio.

C. Sistema Gerenciador Para Clínica Animal

O projeto, desenvolvido por Letícia L. Batista e Paulo César dos Santos [14], demonstra o planejamento e a execução das etapas de desenvolvimento de um *software* para gerenciamento de informações de animais voltados para clínicas veterinárias, possuindo foco no desenvolvimento de uma ferramenta eficiente para auxiliar nas operações diárias da clínica, visando uma gestão mais ágil e organizada. Foi usado como base no desenvolvimento o PHP para o *back-end*, JavaScript, HTML e CSS para o *front-end* e MySQL para o banco de dados. Foi divulgado que o *software* desenvolvido possui a eficiência desejada, atendendo às necessidades da clínica e proporcionando melhoria na gestão e no atendimento aos clientes.

D. Considerações sobre os trabalhos correlatos

Os trabalhos correlatos possuem o foco no desenvolvimento de *softwares* para auxílio dos animais e das entidades envolvidas, buscando eficiência e praticidade, mas sem perder a simplicidade para alcançar a todos os usuários e evitar as “barreiras tecnológicas”. O projeto foi desenvolvido buscando a automação e a melhoria da gestão em clínicas veterinárias, assim como os trabalhos correlatos, utilizando metodologias ágeis garantindo a entrega rápida de funcionalidades e adaptações quando necessário. Além disso, o projeto busca solucionar problemas como a burocracia e os custos relacionados. Visando complementar os trabalhos estudados, o desenvolvimento do projeto adotou tecnologias atuais, como o Spring Boot e o Angular, oferecendo maior escalabilidade, facilidade de manutenção, flexibilidade, segurança e também implementou uma comunicação mais eficiente com os clientes.

IV. METODOLOGIA

Observando as boas práticas aplicadas no planejamento do LocalizaPet, o desenvolvimento do Sistema de Gestão “Pets” também adotou as boas práticas da metodologia *Feature-Driven Development* (FDD) servindo de base para as etapas planejadas. Devido ao grande aumento de profissionais da área, como citado na publicação do VetService, a burocracia envolvida nas atuações tende a ser um obstáculo inicial e contínuo [13], no qual o sistema “Pets” automatiza os processos e reduz a complexidade dos serviços do dia a dia.

No desenvolvimento, foi realizada a aplicação das boas práticas. Foram executadas algumas das etapas realizadas na publicação de Sistema Gerenciador Para Clínica Animal, no qual foi criada uma documentação com complexidade reduzida para melhor entendimento dos envolvidos no desenvolvimento [14] e, posteriormente, nos que irão participar das futuras implementações e manutenções. Para acompanhamento do *stakeholder* foi criado um grupo no WhatsApp para alinhar pontos importantes e validar cenários visando aprimorar ou modificar as funcionalidades do sistema

A. Desenvolvimento do Modelo Abrangente

Visando a eficiência e transmitir melhor o entendimento da aplicação, foi implementado o diagrama de caso de uso para exemplificar os usuários (atores) envolvidos e as funcionalidades base do projeto, como pode ser visto na Figura 1. Este diagrama permite visualizar como os diferentes atores interagem com o sistema e quais são as principais funcionalidades oferecidas. O sistema foi desenvolvido com foco na entrega contínua de funcionalidades, visando garantir a implementação de recursos de maneira sequencial e com testes constantes. Durante o processo de desenvolvimento, foram priorizadas as funcionalidades essenciais para o atendimento na clínica veterinária, com base nas necessidades identificadas no planejamento inicial. O modelo de desenvolvimento adotado permitiu que as entregas fossem feitas de forma ágil e eficaz, resultando em um sistema funcional.

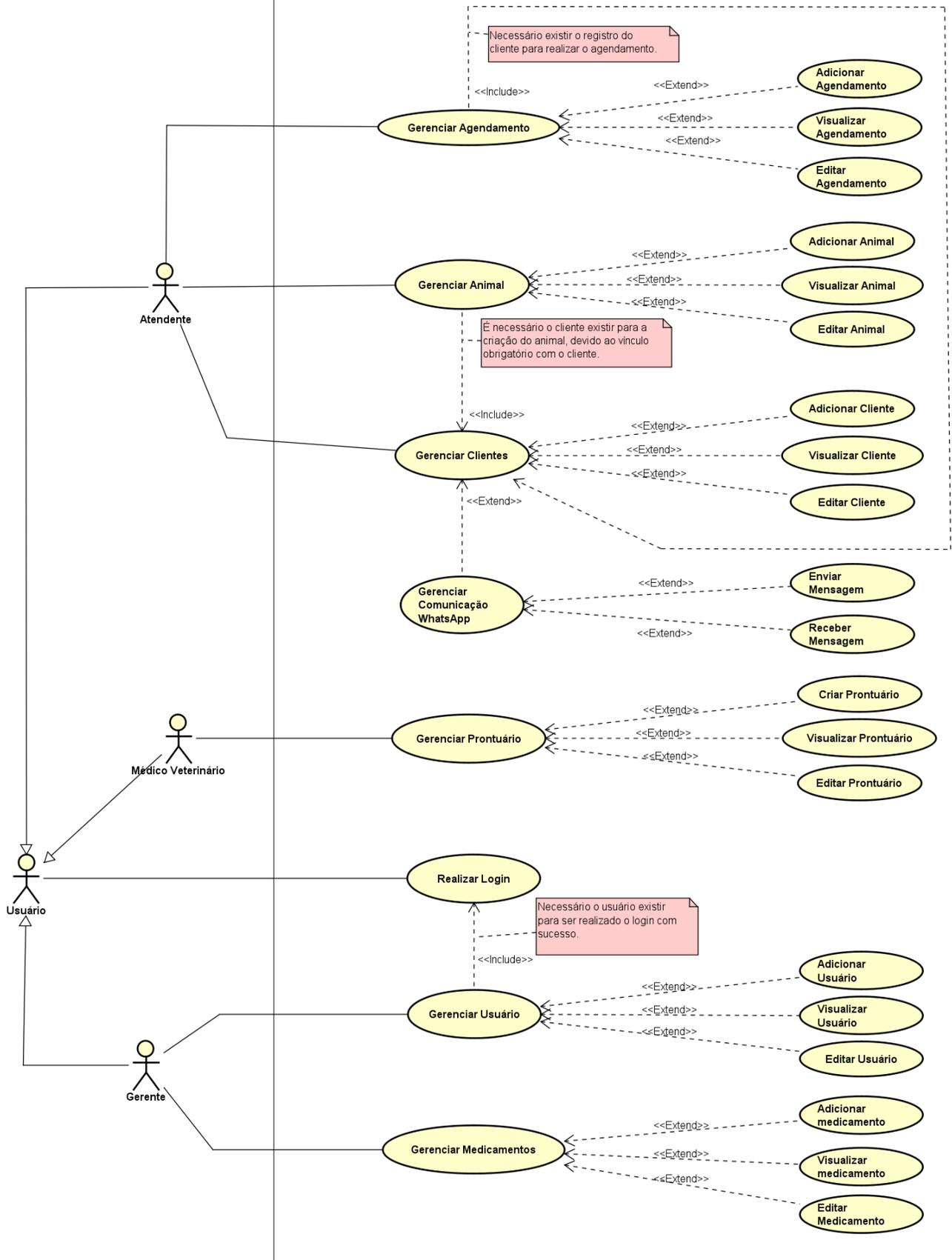


Figura 1: Diagrama de Caso de Uso, Sistema de Gestão "Pets"

B. Lista de funcionalidades

Para estruturar de maneira eficiente as etapas de implementação do projeto, foi elaborada uma lista das funcionalidades essenciais, suas respectivas descrições e a complexidade envolvida em sua implementação. As funcionalidades estão organizadas de acordo com os requisitos funcionais, demonstrado na Figura 2, e requisitos não funcionais, demonstrado na Figura 3.

Funcionalidade	Requisitos Funcionais	Descrição	Complexidade
RF01: Gerenciar usuário	Realiza a manipulação do dado do usuário, efetivando a leitura, a criação e a atualização.	Permite criar, ler e atualizar os dados do usuário.	Média
RF02: Realizar login	Realiza a validação da existência do usuário e permite, ou não, o acesso ao sistema.	Valida se o usuário existe no sistema e permite seu login.	Média
RF03: Gerenciar animal	Realiza a manipulação do dado do animal, efetivando a leitura, a criação e a atualização.	Permite criar, ler e atualizar as informações sobre os animais.	Média
RF04: Gerenciar cliente	Realiza a manipulação do dado do cliente, efetivando a leitura, a criação e a atualização.	Permite criar, ler e atualizar os dados do cliente.	Média
RF05: Gerenciar agendamento	Realiza a manipulação do dado do agendamento, efetivando a leitura, a criação e a atualização.	Permite criar, ler e atualizar os agendamentos realizados pelos clientes.	Média
RF06: Gerenciar prontuário	Realiza a manipulação do dado do prontuário, efetivando a leitura, a criação e a atualização.	Permite criar, ler e atualizar os prontuários dos animais atendidos na clínica veterinária.	Alta
RF07: Gerenciar medicamento	Realiza a manipulação do dado do medicamento, efetivando a leitura, a criação e a atualização.	Permite criar, ler e atualizar os dados sobre os medicamentos administrados aos animais.	Média
RF08: Gerenciar comunicação WhatsApp	Efetiva o envio e o recebimento de mensagens entre o cliente e a clínica veterinária.	Permite enviar e receber mensagens via WhatsApp entre o cliente e a clínica para facilitar a comunicação.	Média

Figura 2: Requisitos Funcionais, Sistema de Gestão “Pets”

Funcionalidade	Requisitos Não Funcionais	Descrição
RNF01	A aplicação deverá ser executada nos navegadores web em geral, permitindo o acesso aos mais variados sistemas operacionais (Windows, MacOS, Linux, Chrome OS, Android, IOS, etc).	A aplicação precisa ser compatível com diversos navegadores e sistemas operacionais, garantindo acessibilidade e usabilidade.
RNF02	A aplicação terá o desenvolvimento realizado com o uso do Spring Boot para o back-end, o Angular para o front-end e o MySQL como gerenciador de banco de dados.	A escolha das tecnologias deve garantir a escalabilidade, eficiência e manutenção da aplicação.

Figura 3: Requisitos Não Funcionais, Sistema de Gestão “Pets”

C. Planejar por funcionalidade

A implementação seguiu a ordem das funcionalidades apresentadas na lista dos requisitos funcionais, como demonstrado na Figura 4. Inicialmente foram implementadas as funcionalidades essenciais, como o gerenciamento de usuário e *login*, garantindo que a base do sistema estivesse sólida. Posteriormente, foram implementadas as funcionalidades mais críticas e complexas, como o gerenciamento de prontuário e o gerenciamento do agendamento, que são fundamentais para o funcionamento eficiente da clínica veterinária.

Funcionalidade	Descritivo	Tempo (horas)
RF01	Gerenciar usuário	14 horas
RF02	Realizar login	14 horas
RF03	Gerenciar animal	14 horas
RF04	Gerenciar cliente	14 horas
RF05	Gerenciar agendamento	14 horas
RF06	Gerenciar prontuário	24 horas
RF07	Gerenciar medicamento	14 horas
RF08	Gerenciar comunicação WhatsApp	14 horas
-	Total	122 horas

Figura 4: Planejamento do desenvolvimento dos requisitos funcionais do Sistema de Gestão “Pets”

Funcionalidades de comunicação, como a integração com o WhatsApp, foram abordadas em fases posteriores, após a conclusão do produto mínimo viável (MVP), assegurando que o sistema principal estivesse estável e funcional antes de adicionar recursos atuais de comunicação. Esse planejamento visou realizar uma implementação progressiva, permitindo testes e ajustes para melhorias contínuas com base nos *feedbacks*.

D. Detalhamento por funcionalidade

Nesta seção, será apresentado o detalhamento das funcionalidades do sistema, demonstrando o modelo de dados e suas interações. Visando uma melhor compreensão do *design* e das interações do sistema, foram desenvolvidos o diagrama de classe, o diagrama de atividade e o diagrama de relacionamento de entidades.

O diagrama de atividade exhibe, de forma clara e objetiva, o fluxo de atividades entre os processos dentro do Sistema de Gestão “Pets”. Sendo desenvolvido para representar a sequência de ações associadas às funcionalidades do sistema, permitindo uma visualização dinâmica das interações.

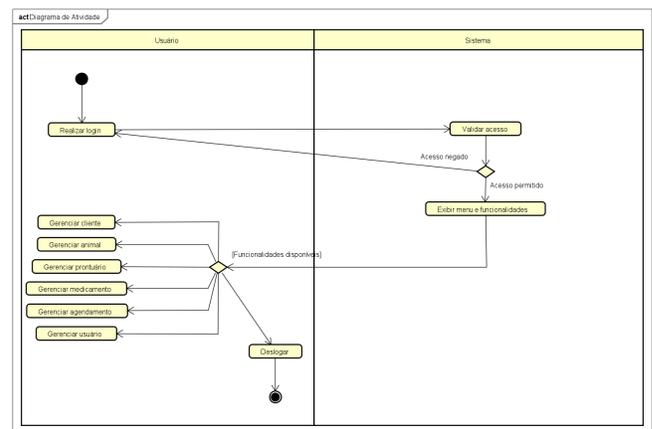


Figura 5: Diagrama de Atividade do Sistema de Gestão “Pets”

O diagrama de classe foi desenvolvido com base nas funcionalidades essenciais do Sistema de Gestão “Pets” e tem como objetivo demonstrar os objetos, as funcionalidades, atributos e relacionamentos do sistema, como ilustrado na Figura 6.

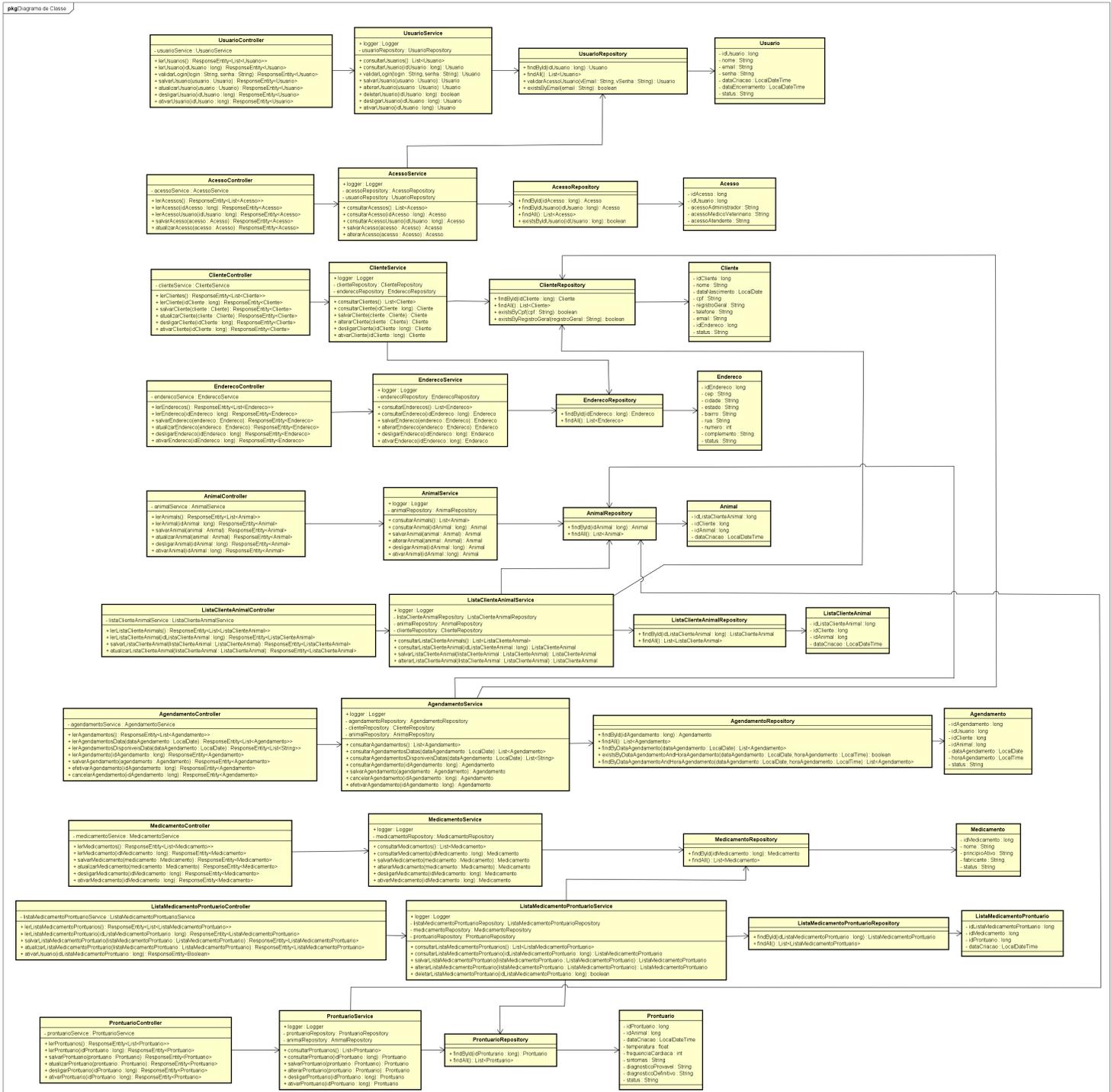


Figura 6: Diagrama de Classe do Sistema de Gestão "Pets"

Para entendimento da estrutura do banco de dados, foi desenvolvido o diagrama de relacionamento de entidades (DER), que ilustra as entidades, seus atributos e os relacionamentos entre elas, permitindo uma melhor compreensão da organização dos dados no sistema, conforme ilustrado na Figura 7.

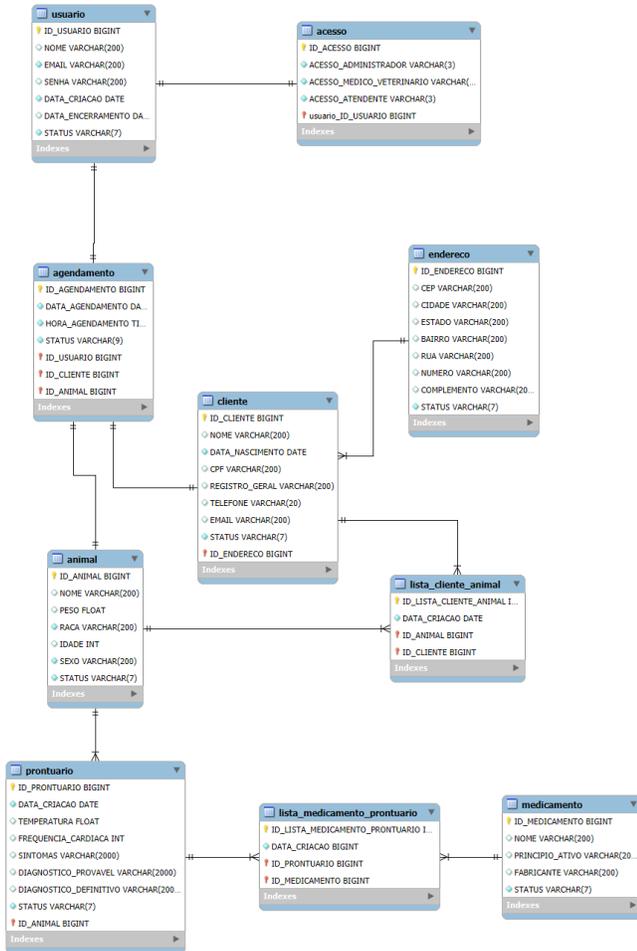


Figura 7: Diagrama de Relacionamento de Entidades do Sistema de Gestão “Pets”

E. Construção por funcionalidade

Nesta seção, será abordada a estrutura da implementação. O desenvolvimento do *back-end*, do banco de dados e do *front-end* foram implementados simultaneamente, no qual seguiu-se a ordem descrita nos requisitos funcionais.

Na Figura 8 está sendo exibida a estrutura do *back-end*, no qual tem início na pasta *src* (*source*). No *controllers* estão os *end-points* que recebem as requisições via RestAPI, no *services* estão os serviços que são chamados pelos *end-points* para processarem as informações e retornarem os dados desejados, no *repositories* estão os repositórios que possuem interfaces do Spring Data JPA que fornecem métodos para manipulação de objetos vinculados a um banco de dados, no *models* estão os modelos (ou classes) do sistema que servem

como entidades para o banco de dados e/ou manipulação nos processamentos dos serviços.

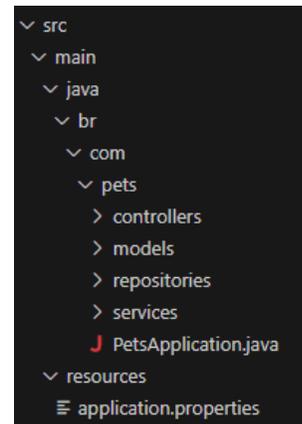


Figura 8: Estrutura do *Back-End* do Sistema de Gestão “Pets”

Na Figura 9 está sendo exibida a estrutura do *front-end*, na qual tem início na pasta *src* (*source*). No *componentes* estão os componentes do Angular que são blocos independentes e reutilizáveis que encapsulam o HTML, o CSS e a lógica relacionada do sistema, no *guards* contém a implementação de guardas de rotas e controlam o acesso aos componentes, no *models* existem os modelos (ou classes) que são usados para mapear os dados entre a aplicação e o banco de dados, no *services* contém os serviços que realizam a comunicação via Rest APIs com o *back-end*.

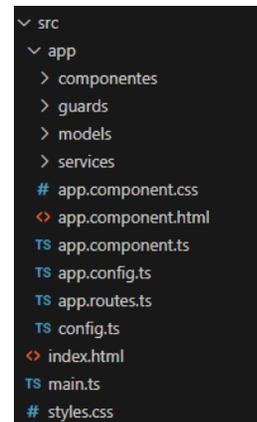


Figura 9: Estrutura do *Front-End* do Sistema de Gestão “Pets”

V. RESULTADOS

Durante o desenvolvimento da aplicação de gestão para clínicas veterinárias, foi retornado desafios relacionados à comunicação com a API do WhatsApp. Embora a API do WhatsApp Business ofereça funcionalidades avançadas, como envio de mensagens automáticas e notificações, sua exigência de um vínculo com um CNPJ para integração inviabilizou seu uso no desenvolvimento do projeto. Para contornar essa limitação, foi optado o vínculo com a API do WhatsApp Web, que apesar de possuir menos funcionalidades chegou a atender

às expectativas iniciais do sistema, garantindo a comunicação eficiente entre os usuários e os clientes.

Como parte dos resultados alcançados, foi desenvolvida a tela de *login*, conforme demonstrada na Figura 10. Definida como a página inicial do sistema, essa funcionalidade inclui um redirecionamento automático do usuário para a tela de *login*, caso ele tente acessar outra rota sem estar autenticado, assegurando uma experiência de uso mais segura.

A tela de login apresenta um formulário com o título "Login". Há campos para "Usuário:" (contendo "pets@gmail.com") e "Senha:" (com pontos para ocultar). Um botão "Entrar" está posicionado na base do formulário.

Figura 10: Tela de *login* do Sistema de Gestão “Pets”

Após a realização do *login*, o usuário é levado à tela de menu, conforme a Figura 11, onde estão centralizadas as funcionalidades principais do sistema. Essas funcionalidades, posicionadas de forma intuitiva na região esquerda da tela, facilitam o acesso e a navegação entre os diferentes componentes do sistema.

A tela de menu mostra um menu lateral com ícones para "Cliente", "Animal", "Prontuário", "Agendamento", "Medicamento" e "Usuário". O conteúdo principal exibe o título "Cliente" e uma tabela com os seguintes dados:

Nome	Data de Nascimento	CPF	Registro Geral	Telefone	Email	WhatsApp	Editar	Desativar
Cliente Teste	08/09/1997	123.123.123-12	12.123.123-1	(05) 99142-3048	cliente.teste@gmail.com			

Figura 11: Tela de menu do Sistema de Gestão “Pets”

Para o gerenciamento de agendamentos, foi implementada a funcionalidade de exibição de um calendário, conforme a Figura 12. Esta ferramenta permite visualizar e organizar consultas de maneira simples e prática.

A tela de agendamento apresenta um calendário para o mês de Novembro de 2024. O dia 11 está selecionado. Ao lado do calendário, há um formulário de "Reserva" com os seguintes dados:

Cliente/Animal	Hora	Confirmar
Cliente Teste / Brucis	10:00:00	Sim

Figura 12: Tela de agendamento do Sistema de Gestão “Pets”

Para a realização do cadastro de prontuários, foi implementada a tela descrita conforme a Figura 13. Esta ferramenta permite o registro dos dados da consulta realizada.

A tela de cadastro do prontuário contém campos para "Animal:", "Temperatura:", "Presença Cardíaca:", "Sintomas:" e "Medicamentos:". O campo "Medicamentos:" possui subcampos para "Dose", "Frequência" e "Via". Botões "Retornar" e "Adicionar" estão na base da tela.

Figura 13: Tela de cadastro do prontuário do Sistema de Gestão “Pets”

VI. CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Este artigo apresentou o desenvolvimento de uma aplicação *web* voltada para a gestão de clínicas veterinárias sem fins lucrativos.

Os objetivos específicos foram alcançados conforme o planejado, realizando a centralização e a automatização do cadastro de clientes e registro das consultas.

A integração da API do WhatsApp Web permitiu vincular o telefone registrado de cada cliente ao WhatsApp ativo no navegador durante o uso, facilitando a comunicação direta, otimizando a rotina da clínica ao possibilitar a confirmação de consultas na véspera, reduzindo ausências e aproveitando melhor os horários disponíveis. Como resultado, o atendimento se torna mais produtivo, permitindo aos veterinários dedicar mais tempo as consultas ou aos animais internados na clínica.

Visando a evolução da aplicação foi observado que, para trabalhos futuros, o desenvolvimento de uma gestão de pagamentos agregaria de forma significativa, no qual possibilitaria a automação de processos financeiros, como funcionalidades direcionadas a geração de relatórios dos gastos e notificações de vencimentos dos pagamentos, possibilitando um gerenciamento mais transparente e organizado, permitindo que as clínicas ofereçam um serviço mais completo e eficiente.

REFERÊNCIAS

- [1] R. P. d. Meneses, G. S. Cantarelli, and G. C. Kurtz, “Localizapet: Desenvolvimento de um sistema web para localização de animais perdidos, doações para ongs e adoção de animais,” https://www.tfgonline.lapinf.ufn.edu.br/media/midias/TFG2_-_LocalizaPet.pdf, 2023, acessado em: 01 de abril de 2024.
- [2] Conselho Federal de Medicina Veterinária (CFMV), “Censo de transparência 2017-2020,” <https://www.cfmv.gov.br/censo/transparencia/2017-2020/2020/12/11/#:~:text=Em%20dezembro%20de%202017%2C%20o,de%2035%2C9%20mil%20profissionais.,> 2020, acessado em: 08 de maio de 2024.
- [3] U. E. Kids, “Animais — ensino fundamental i,” <https://escolakids.uol.com.br/ciencias/os-animais.htm#:~:text=Os%20animais%20s%C3%A3o%20seres%20vivos,a%C3%A9reos%2C%20entre%20v%C3%A1rias%20outras%20formas.,> 2024, acessado em: 30 de abril de 2024.
- [4] Conselho Regional de Medicina Veterinária do Distrito Federal (CRMV-DF), “Resolução define funcionamento de consultórios, clínicas e hospitais,” <https://www.crmvdf.org.br/resolucao-define-funcionamento-de-consultorios-clinicas-e-hospitais/>, 2024, acessado em: 30 de abril de 2024.

- [5] Oracle, “O que é java?” https://www.java.com/pt-BR/download/help/whatis_java.html, 2024, acessado em: 30 de abril de 2024.
- [6] Oracle Corporation, “Java;” <https://www.oracle.com/br/java/>, 2024, acessado em: 30 de abril de 2024.
- [7] Amazon Web Services, “What is an api?” <https://aws.amazon.com/pt/what-is/api/>, 2024, acessado em: 09 de maio de 2024.
- [8] WhatsApp, “Sobre o whatsapp;” <https://www.whatsapp.com/about>, 2024, acessado em: 21 de maio de 2024.
- [9] WhatsApp API, “Whatsapp business api;” <https://developers.facebook.com/docs/whatsapp/cloud-api/overview>, 2024, acessado em: 21 de maio de 2024.
- [10] IBM, “Java spring boot;” <https://www.ibm.com/br-pt/topics/java-spring-boot>, 2024, acessado em: 30 de abril de 2024.
- [11] Angular Team, “What is angular?” <https://angular.io/guide/what-is-angular>, 2024, acessado em: 30 de abril de 2024.
- [12] Oracle Corporation, “What is mysql?” <https://www.oracle.com/br/mysql/what-is-mysql/>, 2024, acessado em: 30 de abril de 2024.
- [13] E. L. Chaves and G. G. Rezende, “Vetservice: Sistema de gestão para clínicas veterinárias;” https://ric.cps.sp.gov.br/bitstream/123456789/11277/1/analisedesenvolvimentosistema_2022_2_elisergustavo_vetservicesistemadegestaoparaclinicasveterinarias_TG.pdf, 2022, acessado em: 01 de junho de 2024.
- [14] L. da Silva Souza and P. H. de Oliveira, “Sistema gerenciador para clínica animal;” <https://josif.ifsuldeminas.edu.br/ojs/index.php/anais/article/view/1040/787>, 2023, acessado em: 01 de junho de 2024.
- [15] P. J. Deitel and H. M. Deitel, Como programar em Java, 10th ed. Editora Deitel, 2018.
- [16] Spring Boot Team, “Spring boot reference documentation;” <https://docs.spring.io/spring-boot/docs/current/reference/html/>, 2024, acessado em: 19 de março de 2024.