Documentação do projeto VetTrack

Autores: **Franciele Dalarosa e Jonathan Gotz Correa**

2024

ÍNDICE DETALHADO

Prefácio

O objetivo deste documento é apresentar a especificação do sistema VetTrack, um software para gerenciamento de consultas em clínicas veterinárias. A documentação foi elaborada com base nos princípios de Engenharia de Software, utilizando a notação UML para facilitar a compreensão dos requisitos e design do sistema..

Modelo da Documentação

Esta seção descreve o roteiro utilizado para documentar o desenvolvimento do sistema VetTrack, abrangendo desde a fase inicial de levantamento de requisitos até a implantação do sistema em ambiente de produção.

O processo de documentação foi orientado a objetos e utiliza a notação UML (Unified Modeling Language) para assegurar clareza e consistência. Este modelo é composto pelas seguintes partes: Requisitos:

1. **Requisitos:** Descrição dos requisitos funcionais e não funcionais do sistema, baseados nas necessidades identificadas nas clínicas veterinárias.

2. **Modelagem UML:** Utilização de diagramas UML, como diagramas de casos de uso, classes, atividades e estados, para representar a estrutura e o comportamento do sistema.

3. **Protótipos e Interface:** Apresentação de protótipos que representam a interface preliminar do sistema, detalhando telas e interações com os usuários.

4. **Arquitetura do Sistema:** Descrição da arquitetura cliente-servidor e das tecnologias empregadas, como React.js, Flutter e PostgreSQL.

5. **Modelo de Dados:** Criação do modelo lógico do banco de dados, com tabelas e relacionamentos derivados do diagrama de classes.

6. **Testes:** Definição de um plano de testes detalhado para validar funcionalidades, segurança e desempenho do sistema.

7. **Implantação:** Diretrizes para instalação e configuração do sistema, assegurando que ele esteja operacional nas clínicas.

Cada seção foi organizada para fornecer uma visão clara e objetiva das etapas do desenvolvimento, servindo como referência para equipes técnicas e stakeholders do projeto.

1. Introdução ao Documento

Este documento apresenta especificações do VetTrack, um sistema de gerenciamente de consultas e atendimentos para clínicas veterinárias, desenvolvido para atender às necessidades de controle e otimização de processos clínicos e administrativos. O sistema é projetado para registrar, gerenciar e monitorar consultas, diagnósticos, relatórios e informações de pacientes e tutores.

O desenvolvimento baseia-se nas melhores práticas de engenharia de software, com modelagem UML para assegurar clareza na estrutura e funcionalidade.

1.1. Tema

O projeto aborda o desenvolvimento de um Sistema de Gestão para Clínicas Veterinárias, com foco na organização e automação dos processos de consulta e diagnóstico. Este sistema destina-se a suportar as atividades diárias dos profissionais da clínica, oferecendo recursos de gerenciamento de informações sobre consultas, exames, cadastro de pacientes e comunicação com tutores.

1.2. Objetivo do Projeto

O projeto tem como objetivo geral desenvolver um Sistema de Gestão de Consultas e Atendimentos Veterinários que permita gerenciar com eficiência os dados de tutores, pets, veterinários e consultas em uma clínica veterinária.

1.3. Delimitação do Problema

O problema a ser resolvido foca na gestão de informações clínicas e administrativas em clínicas veterinárias de pequeno e médio porte. Muitas clínicas ainda utilizam processos manuais ou sistemas desatualizados, o que dificulta o controle eficiente de informações e pode comprometer a qualidade do atendimento ou perda de informações cruciais.

O projeto se delimita ao desenvolvimento de um sistema capaz de gerenciar apenas consultas, diagnósticos, agendamentos e comunicação com os tutores dos pets. O sistema não abrange funcionalidades relacionadas a estoque de medicamentos, faturamento, ou controle financeiro da clínica. A proposta específica do projeto é concentrar-se na organização dos dados clínicos e de consulta, promovendo uma gestão mais ágil e integrada entre veterinários e funcionários da clínica.

1.4. Justificativa da Escolha do Tema

A escolha do tema deste projeto foi motivada pela necessidade prática de apoiar o trabalho de profissionais veterinários, especialmente em clínicas de médio e pequeno porte, onde a organização e a eficiência nos atendimentos podem enfrentar desafios devido a recursos limitados ou processos manuais. Este cenário foi observado de perto devido à experiência direta da namorada do desenvolvedor, que trabalha como veterinária e lida com essas dificuldades cotidianamente.

1.5. Método de Trabalho

O desenvolvimento do projeto seguirá uma abordagem orientada a objetos com javascript, aplicando metodologias ágeis para permitir um desenvolvimento incremental e adaptativo. O sistema será implementado utilizando React.js para o frontend, proporcionando uma interface responsiva e dinâmica para a interação do usuário. Para o desenvolvimento do aplicativo móvel, será utilizado Flutter, que oferece uma interface nativa e eficiente para dispositivos móveis.

Além disso, o sistema integrará um modelo de inteligência artificial dedicado a capturar e anotar informações relevantes durante as consultas veterinárias. Esse modelo permitirá que o veterinário tenha suporte para documentar diagnósticos e observações por meio de anotações automáticas em tempo real, otimizando o tempo e a precisão na coleta de dados durante os atendimentos.

[1] Para maiores detalhes dos tipos de processos de desenvolvimento de software consultar o livro Engenharia de Software – Roger Pressman – 5ª edição - Capítulo 2.

1.6. Organização do Trabalho

Este documento foi estruturado para abranger todas as etapas do desenvolvimento do sistema VetTrack, desde a concepção inicial até sua implantação e considerações finais. A organização segue uma abordagem lógica e sequencial, facilitando a consulta por parte de desenvolvedores, usuários e outros interessados.

1.7. Glossário

A

Administrador: Usuário responsável por gerenciar permissões, usuários do sistema e configurações gerais.

Agendamento: Processo de marcar um encontro (consulta, exame ou cirurgia) no sistema.

B

Banco de Dados: Repositório de informações estruturadas, utilizado para armazenar dados de tutores, pets, veterinários e encontros no sistema VetTrack.

C

Cadastro: Ato de registrar informações de um tutor, pet, usuário ou encontro no sistema.

Caso de Uso: Representação de uma interação específica entre um ator (usuário) e o sistema, descrita em diagramas UML.

Cliente: Dispositivo usado para acessar o sistema VetTrack, seja um computador ou smartphone.

Consulta: Tipo de encontro no qual o veterinário atende um pet para avaliação ou diagnóstico.

D

Diagrama de Classe: Representação gráfica das classes do sistema, seus atributos, métodos e relacionamentos.

Diagrama de Sequência: Representação das interações temporais entre objetos ou componentes do sistema.

E

Encontro: Termo usado no VetTrack para definir eventos agendados, como consultas, exames e cirurgias.

F

Frontend: Parte do sistema visível e interativa para o usuário, desenvolvida com React.js no VetTrack.

I

Interface: Tela ou conjunto de telas do sistema utilizadas para interação com o usuário.

Inteligência Artificial (IA): Tecnologia integrada ao VetTrack para auxiliar os veterinários com anotações automáticas e sugestões de diagnósticos.

M

Middleware: Componente do backend responsável por processar solicitações antes que sejam atendidas pelo sistema principal, como autenticação de usuários.

P

Pet: Animal cadastrado no sistema, associado a um tutor.

PostgreSQL: Banco de dados relacional utilizado no VetTrack para armazenamento de informações.

R

Relatório: Documento gerado pelo sistema contendo informações organizadas, como histórico de encontros e dados de pets.

Requisito Funcional: Requisitos que descrevem o que o sistema deve realizar para atender às necessidades dos usuários.

Requisito Não Funcional: Requisitos que especificam restrições ou características de qualidade do sistema, como desempenho e segurança.

S

Segurança: Conjunto de práticas e mecanismos para proteger os dados sensíveis armazenados e processados pelo sistema.

Servidor: Máquina responsável por hospedar o backend e gerenciar as requisições enviadas pelos clientes.

T

Tutor: Pessoa responsável por um ou mais pets cadastrados no sistema.

U

Usuário: Qualquer pessoa que interage com o sistema VetTrack, como veterinários, funcionários administrativos e administradores.

V

Veterinário: Profissional que realiza consultas e procedimentos no sistema VetTrack, utilizando recursos como o assistente virtual.

2. Descrição Geral do Sistema

O sistema VetTrack é uma plataforma orientada a objetos desenvolvida para facilitar o atendimento em clínicas veterinárias de pequeno e médio porte. Ele abrange funcionalidades essenciais, como agendamento e gerenciamento de consultas, cadastro de tutores e pets, registro de diagnósticos e prescrições. O sistema também integra um assistente virtual que realiza anotações automáticas durante as consultas, permitindo ao veterinário registrar dados de forma eficiente. Com uma interface amigável e acessível, o sistema otimiza processos, melhora a organização e assegura uma experiência prática para o gerenciamento clínico.

2.1. Descrição do Problema

O sistema de gestão de consultas veterinárias visa resolver problemas comuns em clínicas veterinárias de médio e pequeno porte, onde o fluxo de informações e o gerenciamento de dados dos pacientes e consultas são muitas vezes descentralizados e ineficientes.

- ✓ Quem é afetado pelo sistema?

  O sistema afeta diretamente veterinários e funcionários administrativos que gerenciam as consultas e o histórico médico dos pacientes. Além disso, os tutores dos animais são impactados de forma indireta, uma vez que o sistema proporcionará um atendimento mais organizado e eficaz.

- ✓ Qual é o impacto do sistema?

  A implementação do sistema permitirá que as clínicas tenham maior controle sobre as consultas, prontuários, agendamentos e diagnósticos, tornando os processos mais ágeis e menos suscetíveis a erros humanos. A centralização das informações e a possibilidade de incluir um assistente virtual durante a consulta representam uma melhora significativa na precisão dos registros e na satisfação dos tutores.

- ✓ Qual seria uma boa solução para o problema?

  Uma solução eficiente seria o desenvolvimento de um sistema de gestão baseado em uma arquitetura orientada a objetos, com módulos específicos para o cadastro de pacientes, agendamentos, controle de prontuários, e uma ferramenta de apoio para anotações durante as consultas, auxiliada por inteligência artificial. Essa solução ajudará na organização dos dados, melhorando a produtividade da equipe e garantindo uma experiência de atendimento mais satisfatória para os tutores e seus animais.

2.2. Principais Envolvidos e suas Características

2.1.1. Usuários do Sistema

Veterinário: profissionais responsáveis por realizar consultas e diagnósticos. Utilizarão o sistema para acessar informações de pacientes, registrar anotações e gerar relatórios.

Funcionários administrativos (usuário clínica): Colaboradores que gerenciam o agendamento de consultas, cadastro de tutores e pets, além de manter o funcionamento diário da clínica.

Administradores: Colaboradores que gerenciam cadastro de usuários e dados críticos do sistema.

2.1.2. Desenvolvedores do Sistema

Desenvolvedores: Programadores que implementarão o sistema utilizando tecnologias como React.js e Flutter, assegurando que as funcionalidades sejam desenvolvidas de acordo com as especificações.

2.3. Regras de Negócio

As regras de negócio do VetTrack definem as diretrizes e restrições essenciais para o funcionamento eficiente e seguro do sistema em uma clínica veterinária. Essas regras incluem:

Restrição de Acesso por Nível de Usuário: O sistema possui diferentes níveis de acesso para funcionários administrativos e veterinários, garantindo que informações sensíveis sejam acessadas apenas por pessoas autorizadas.

Confirmação de Agendamento: Todas as consultas devem ser confirmadas antes de serem registradas no sistema, garantindo que o veterinário esteja disponível e que o tutor tenha ciência do horário e das condições do atendimento.

Restrições de Desempenho: O sistema deve responder em tempo real às ações do usuário, principalmente durante o uso do assistente virtual para anotações, garantindo que a experiência do usuário seja fluida e sem interrupções.

Volume de Dados e Armazenamento: Espera-se que o sistema armazene um volume crescente de informações sobre tutores, pets e históricos de consultas. As estimativas iniciais indicam um aumento anual de aproximadamente 20% no volume de dados, exigindo capacidade de escalabilidade no banco de dados.

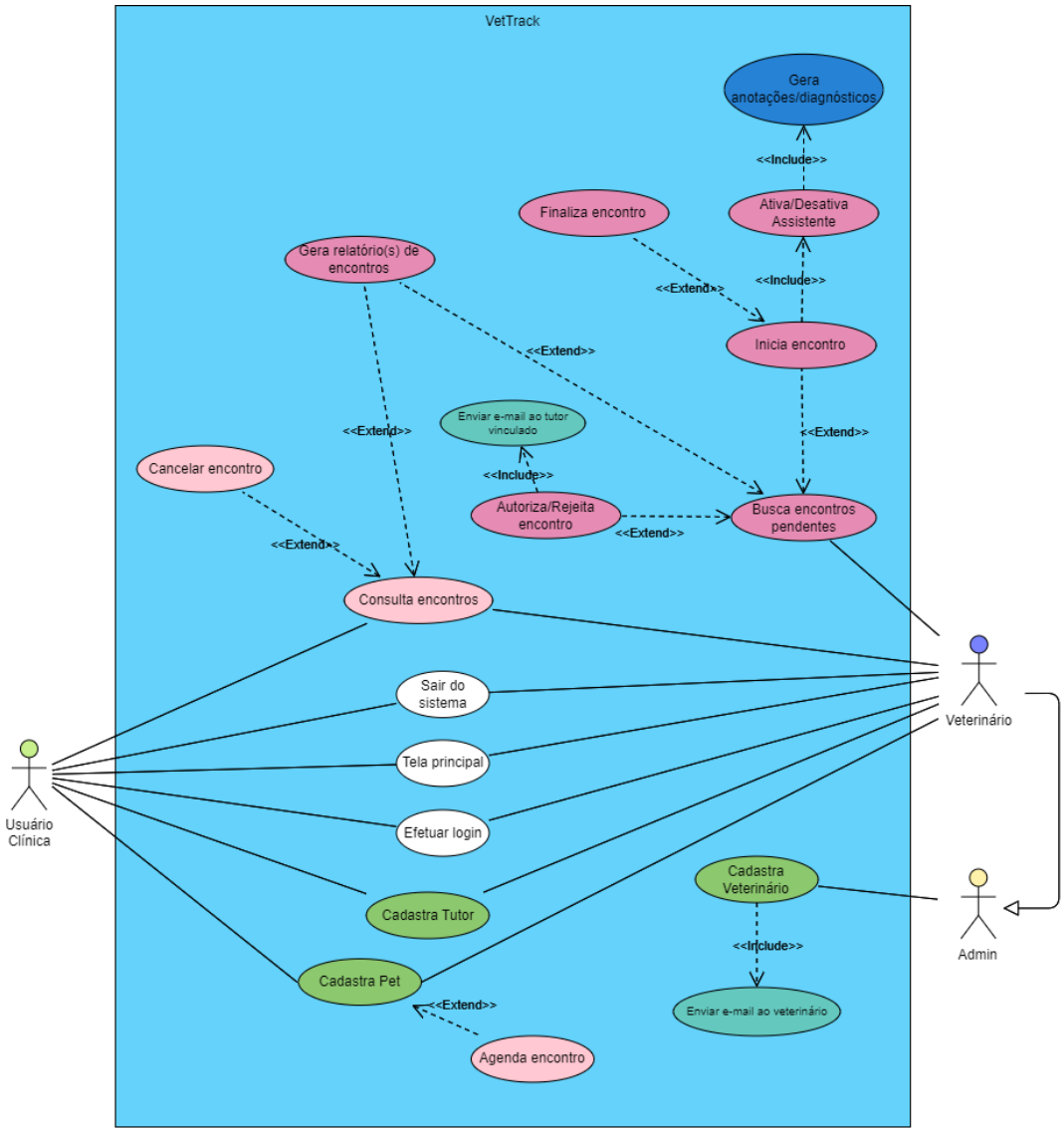
Regras de Atualização de Histórico: Apenas veterinários e administradores têm permissão para editar ou corrigir informações no histórico de consultas ou diagnósticos após o encerramento de um atendimento.

Ferramentas de Apoio: O sistema utilizará um modelo de inteligência artificial para auxiliar na coleta e organização de informações, suportado por um microfone que permitirá o reconhecimento de voz para anotações automáticas..

3. Requisitos do Sistema

Este capítulo tem como objetivo descrever os requisitos do sistema.

3.1. Requisitos Funcionais



- **RF02:** Cadastro de Veterinário
- **Breve descrição:** Permitir que o administrador cadastre novos veterinários.
- **Atores envolvidos:** Administrador.
- **Pré-condições:** O administrador deve estar autenticado no sistema.

- **Seqüência de Eventos ou Fluxo Principal de Eventos:** O administrador acessa a funcionalidade de cadastro de veterinário. Insere as informações obrigatórias (nome, CPF, e-mail, telefone, idade, área de atuação, data de início dos serviços, número da carteira de medicina). Confirma o cadastro, o sistema valida os dados e registra o veterinário e o sistema envia um e-mail com as credenciais de acesso.
- **Pós-condições:** O veterinário está registrado no sistema com suas credenciais enviadas.
- **Exceções ou Fluxo Secundário de Eventos:** Dados inválidos ou CPF/e-mail duplicado.
- **Observações:** A senha gerada será aleatória e o veterinário deverá alterá-la no primeiro login.

- **RF03 e RF04:** Cadastro de Tutor e Cadastro de Pet
- **Breve descrição:** Permitir que os usuários cadastrem tutores e o cadastro de pets.
- **Atores envolvidos:** Todos.
- **Pré-condições:** Tutor: O usuário deve estar autenticado no sistema.

  Pet: O tutor deve estar cadastrado previamente.
- **Seqüência de Eventos ou Fluxo Principal de Eventos:** O usuário acessa a funcionalidade de cadastro de tutor/pet, insere as informações obrigatórias (para o cadastro do tutor é o nome, CPF, e-mail, telefone, enquanto do pet são nome, raça, espécie e idade) e opcionais (tutor: idade e pet: tipo sanguíneo e peso) e confirma o cadastro, então o sistema valida os dados e registra o tutor/pet.
- **Pós-condições:** Tutor: O tutor está registrado no sistema.

  Pet: O pet está registrado no sistema vinculado a um tutor.
- **Exceções ou Fluxo Secundário de Eventos:** Tutor: Dados inválidos ou CPF/e-mail duplicado. Pet: Tutor inexistente ou dados do pet inválidos.
- **Observações:** O vínculo entre pet e tutor é essencial para agendamentos

---

- **RF05:** Agendamento de Encontro
- **Breve descrição:** Permitir o agendamento de encontros.
- **Atores envolvidos:** Todos.
- **Pré-condições:** O tutor e o pet devem estar cadastrados no sistema.
- **Seqüência de Eventos ou Fluxo Principal de Eventos:** O usuário acessa a funcionalidade de agendamento, seleciona o pet e insere as informações do encontro (motivo, data, hora, descrição, tutor). O sistema verifica disponibilidade para o horário solicitado, então, confirma o agendamento como "pendente de autorização".
- **Pós-condições:** O encontro está agendado e aguardando autorização.
- **Exceções ou Fluxo Secundário de Eventos:** Horário indisponível ou informações inválidas.
- **Observações:** O tutor será notificado automaticamente por e-mail.

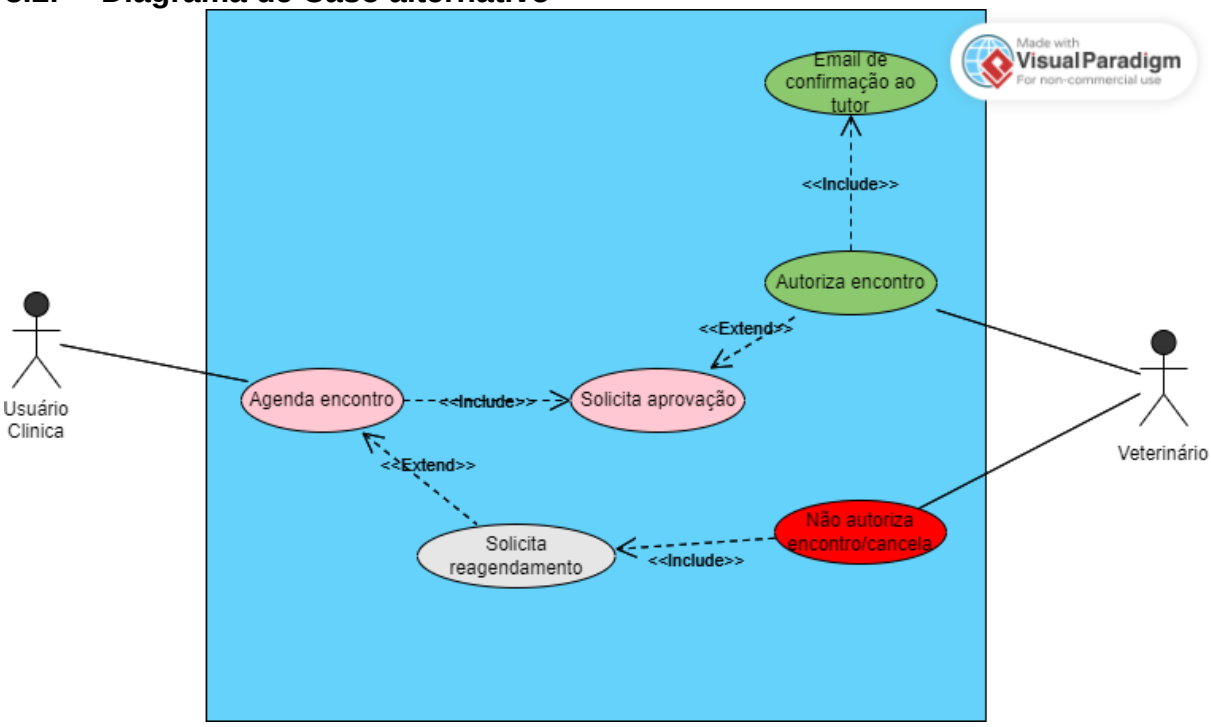
- **RF06:** Autorização de Encontro
- **Breve descrição:** Permitir a autorização de encontros pendentes.
- **Atores envolvidos:** Veterinário.
- **Pré-condições:** O encontro deve estar pendente de autorização.
- **Seqüência de Eventos ou Fluxo Principal de Eventos:** O veterinário acessa a lista de encontros pendentes, consulta o resumo das informações do encontro, e então, autoriza ou rejeita o encontro.

  O sistema registra a decisão e notifica o tutor por e-mail.
- **Pós-condições:** O status do encontro é atualizado para autorizado ou rejeitado.
- **Exceções ou Fluxo Secundário de Eventos:** Falha na comunicação com o tutor.
- **Observações:** Apenas veterinários têm permissão para autorizar encontros.

- **RF08:** Ativar/Desativar Assistente Virtual
- **Breve descrição:** Permitir ativar ou desativar o assistente virtual.
- **Atores envolvidos:** Veterinário.
- **Pré-condições:** A consulta deve estar em andamento.
- **Seqüência de Eventos ou Fluxo Principal de Eventos:** O veterinário seleciona a opção de ativar/desativar o assistente, o sistema ativa ou desativa o reconhecimento de voz inicia-se a transcrição simultanea de informações relevantes segundo o modelo de I.A utilizada.
- **Pós-condições:** O estado do assistente virtual é alterado.
- **Exceções ou Fluxo Secundário de Eventos:** Microfone indisponível ou falha no reconhecimento de voz.
- **Observações:** O assistente virtual melhora a produtividade durante consultas.

- **RF09:** Relatório de Encontros
- **Breve descrição:** Permitir gerar relatórios filtrados sobre os encontros.
- **Atores envolvidos:** Todos.
- **Pré-condições:** O sistema deve conter dados de encontros registrados.
- **Seqüência de Eventos ou Fluxo Principal de Eventos:** O usuário acessa a funcionalidade de relatórios, seleciona os filtros desejados (data, tipo de consulta, etc.) e, então, gera o relatório.
- **Pós-condições:** O relatório é exibido ou exportado.
- **Exceções ou Fluxo Secundário de Eventos:** Falha ao aplicar filtros ou ausência de dados.
- **Observações:** Relatórios podem ser exportados em PDF ou Excel.

3.2. Diagrama de Caso alternativo



3.3. Requisitos Não-Funcionais

ID	Requisito não Funcional	Descrição
RNF01	Segurança	Dados sensíveis devem ser criptografados.
RNF02	Usabilidade	Responsivo às telas de dispositivos (celular, tablet, navegador).
RNF03	Usabilidade	A interface deve ser intuitiva com opções claras para visualizar menu e fazer pedidos.
RNF04	Desenvolvimento/ Implementação	Banco de dados: PostgreSQL.
RNF05	Disponibilidade	O sistema deve estar disponível 7 dias por semana, 24 horas por dia, exceto em manutenções programadas.
RNF06	Manutenibilidade	Deve ser fácil adicionar novos itens ao menu, sem a necessidade de intervenção de um desenvolvedor.
RNF07	Manutenibilidade	O código-fonte deve ser organizado e documentado de acordo com as melhores práticas.
RNF08	Legislação e Regulamentação	Deve cumprir as legislações locais e nacionais, relacionadas a vendas de bebidas alcoólicas.

3.4. Métricas e Cronograma

Estimativa de Esforço

- **Casos de Uso Simples**: Cadastro de tutores e pets.
  - Pontos estimados: 5 UCP por caso.
- **Casos de Uso Médios**: Agendamento de consultas e geração de relatórios.
  - Pontos estimados: 10 UCP por caso.
- **Casos de Uso Complexos**: Integração com assistente virtual e controle de permissões de usuários.
  - Pontos estimados: 15 UCP por caso.

Total de Pontos de Caso de Uso: 65 UCP.
Fator de Conversão: 20 horas por UCP (estimativa média).
Esforço Total: 1300 horas.

Alocação de Recursos

Os recursos foram distribuídos considerando as seguintes categorias:

- **Desenvolvedores Frontend (React.js)**: 2 desenvolvedores alocados.
- **Desenvolvedores Mobile (Flutter)**: 1 desenvolvedor alocado.
- **Desenvolvedores Backend e Banco de Dados (Node.js e PostgreSQL)**: 2 desenvolvedores alocados.

- **Especialista em IA (Assistente Virtual)**: 1 especialista alocado.

Cronograma

O cronograma foi elaborado considerando uma abordagem incremental e adaptativa, conforme descrito na metodologia ágil adotada.

Tarefa	Duração Estimada	Responsável	Início	Término
Levantamento de Requisitos	2 semanas	Equipe de Análise	01/01/2024	14/01/2024
Design de Arquitetura e Protótipos	3 semanas	Equipe de Design	15/01/2024	04/02/2024
Desenvolvimento Frontend	4 semanas	Desenvolvedores Frontend	05/02/2024	03/03/2024
Desenvolvimento Backend	5 semanas	Desenvolvedores Backend	05/02/2024	10/03/2024
Integração do Assistente Virtual	3 semanas	Especialista em IA, Backend	11/03/2024	31/03/2024
Testes e Validação	4 semanas	Equipe de QA	01/04/2024	28/04/2024
Implantação e Treinamento	2 semanas	Equipe de Implantação e Suporte	29/04/2024	12/05/2024

4. Análise e *Design*

A análise e o design do sistema VetTrack foram desenvolvidos com base nos requisitos funcionais e não funcionais levantados anteriormente. O objetivo é apresentar uma visão clara da solução técnica proposta, incluindo a arquitetura do sistema e os diagramas que modelam sua estrutura e comportamento.

4.1. Arquitetura do Sistema

O sistema VetTrack utiliza uma arquitetura cliente-servidor de três camadas, composta por:

1. **Camada de Apresentação (Frontend):**

   Desenvolvida em React.js, oferece uma interface web responsiva e amigável.

   Funciona como ponto de interação para usuários administrativos, veterinários e tutores.

2. **Camada de Negócio (Backend):**

   Implementada em Node.js, concentra as regras de negócio do sistema.

   Inclui APIs RESTful para comunicação entre o frontend e o banco de dados.

3. **Camada de Dados (Banco de Dados):**

   Utiliza PostgreSQL para armazenar informações de tutores, pets, consultas e relatórios.

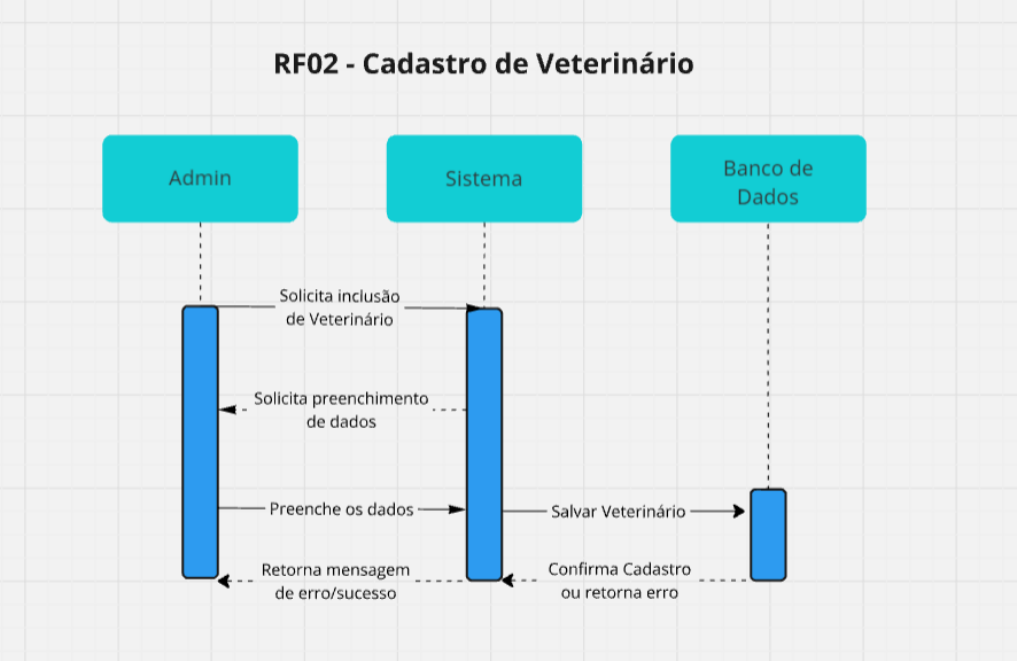
   Foi projetada para atender aos requisitos de escalabilidade e segurança.

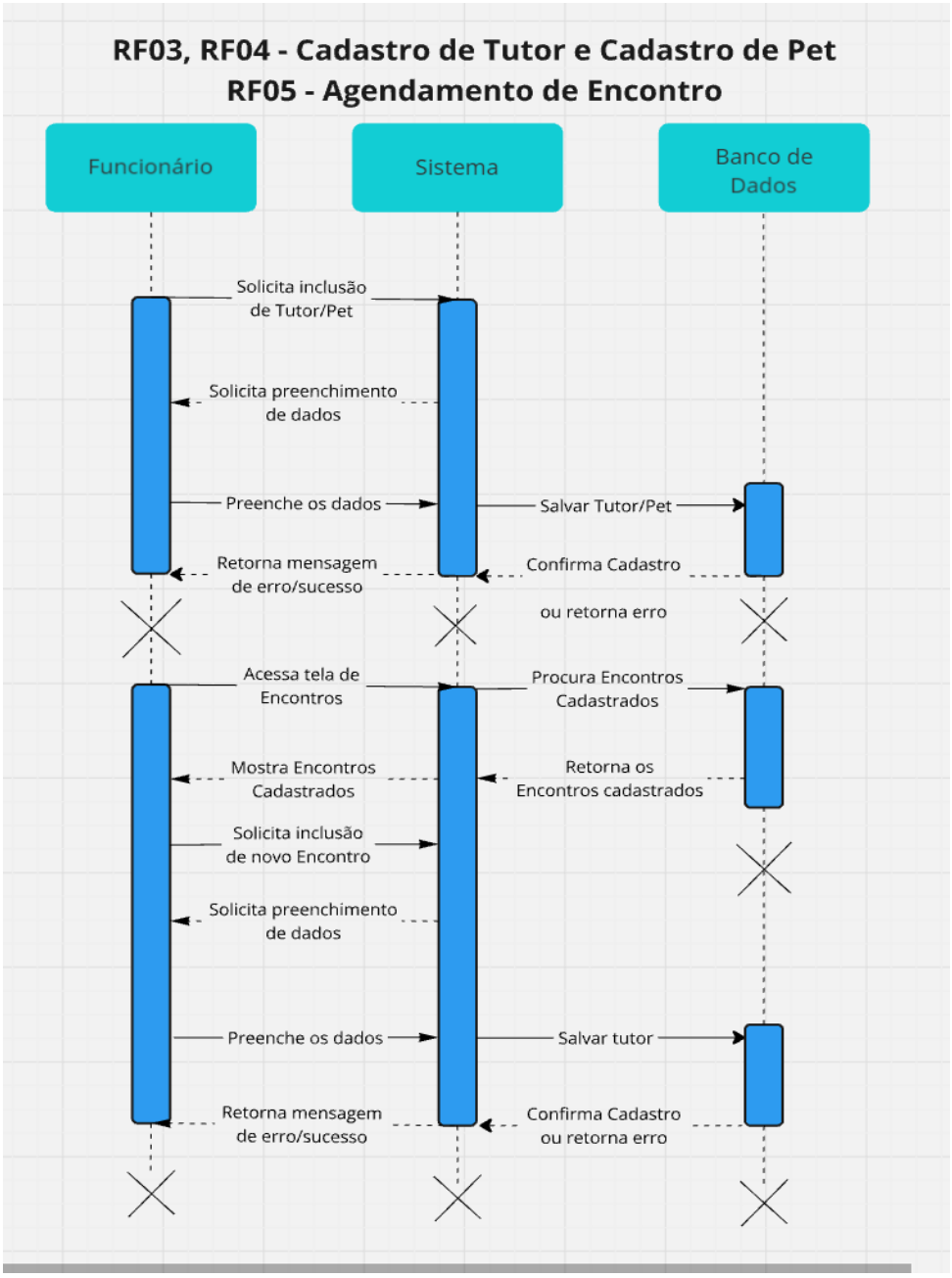
   Adicionalmente, o sistema integra um modelo de inteligência artificial (IA) para o assistente virtual, processando entradas de voz durante as consultas e gerando sugestões de diagnósticos.

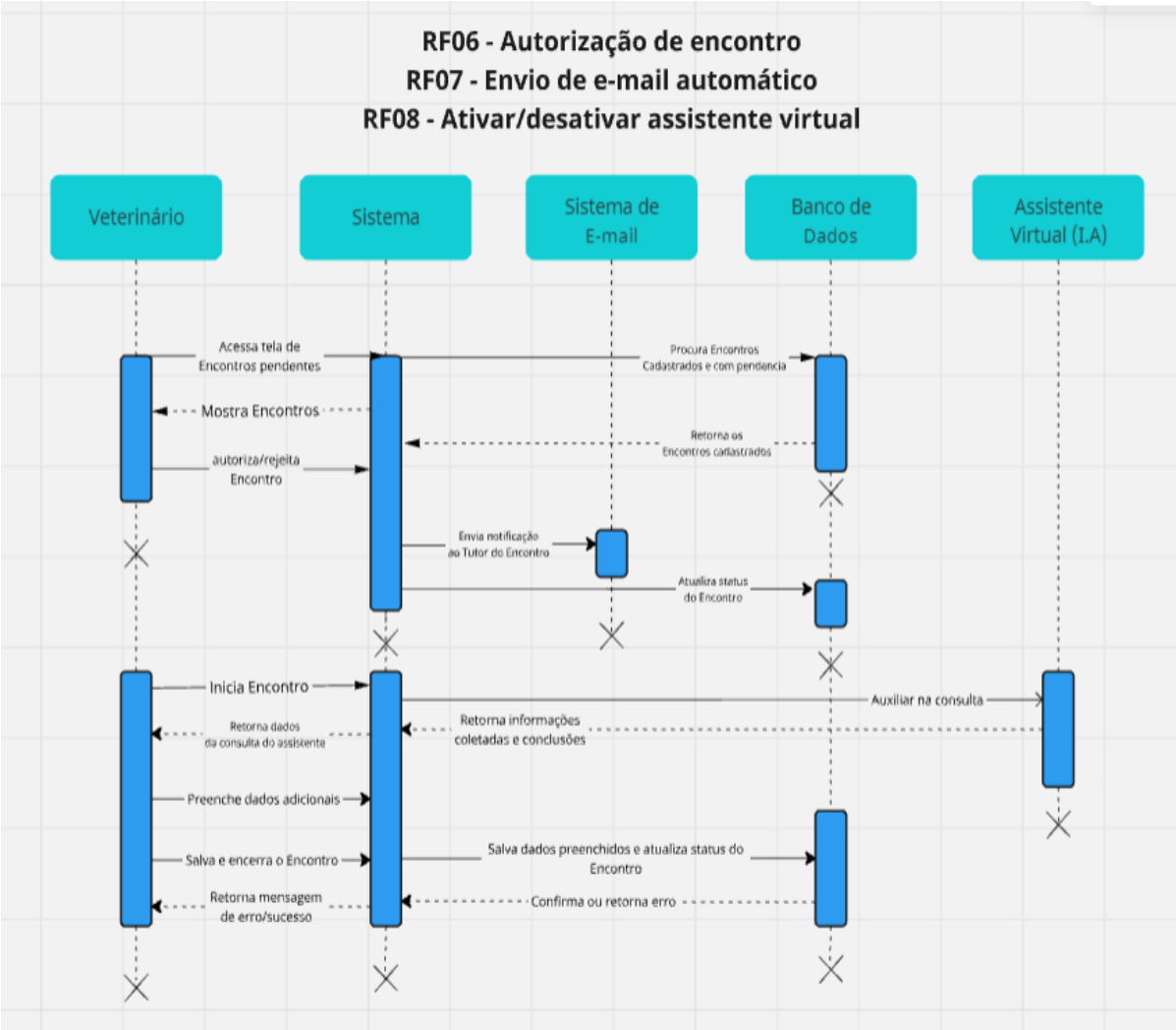
4.2. Diagramas de Interação

O diagrama de interação é composto pelos diagramas de seqüência e colaboração (comunicação, versão 2.0 UML) e modela os aspectos dinâmicos do sistema, mostrando a interação formada por um conjunto de objetos permitindo identificar mensagens que poderão ser enviadas entre eles.

4.3.1. Diagrama de Seqüência

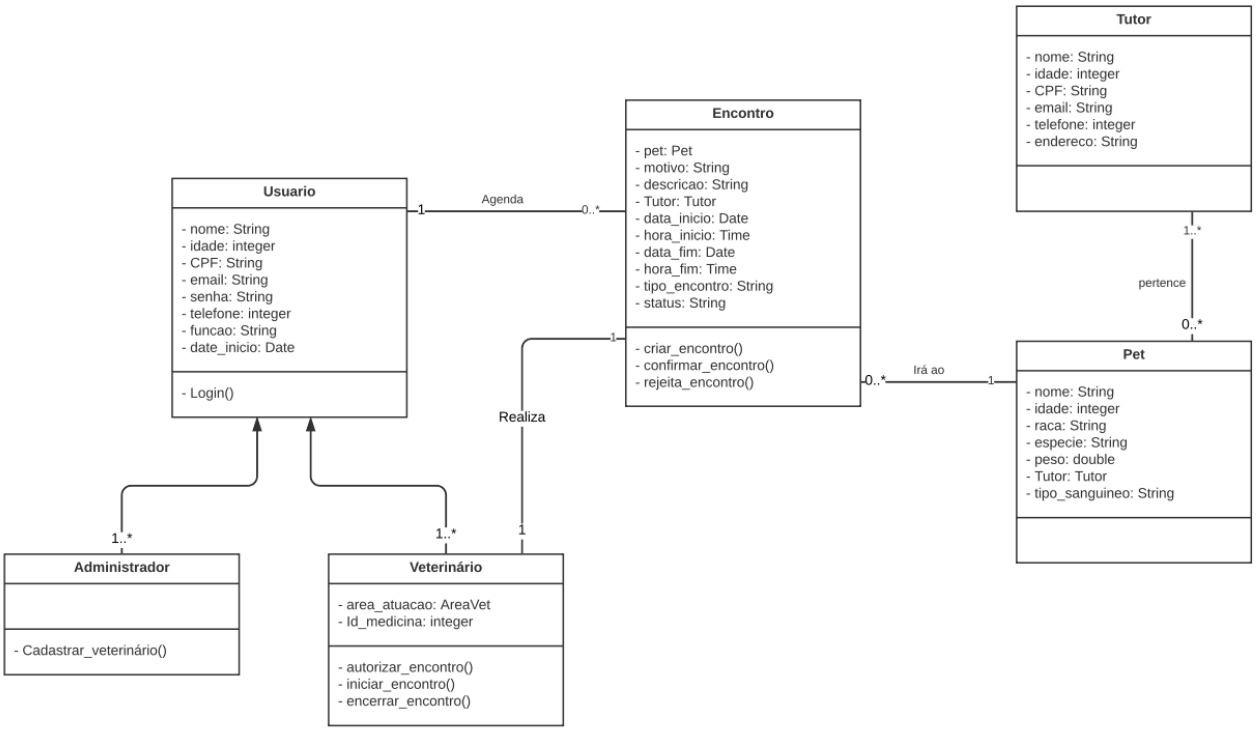



RF03, RF04 - Cadastro de Tutor e Cadastro de Pet
RF05 - Agendamento de Encontro
Funcionário
Sistema
Banco de Dados
Solicita inclusão de Tutor/Pet
Solicita preenchimento de dados
Preenche os dados
Salvar Tutor/Pet
Retorna mensagem de erro/sucesso
Confirma Cadastro ou retorna erro
Acessa tela de Encontros
Procura Encontros Cadastrados
Mostra Encontros Cadastrados
Retorna os Encontros cadastrados
Solicita inclusão de novo Encontro
Solicita preenchimento de dados
Preenche os dados
Salvar tutor
Retorna mensagem de erro/sucesso
Confirma Cadastro ou retorna erro



Notação do diagrama de seqüência.

4.3. Diagrama de Classes



Modela as principais entidades do sistema, como:

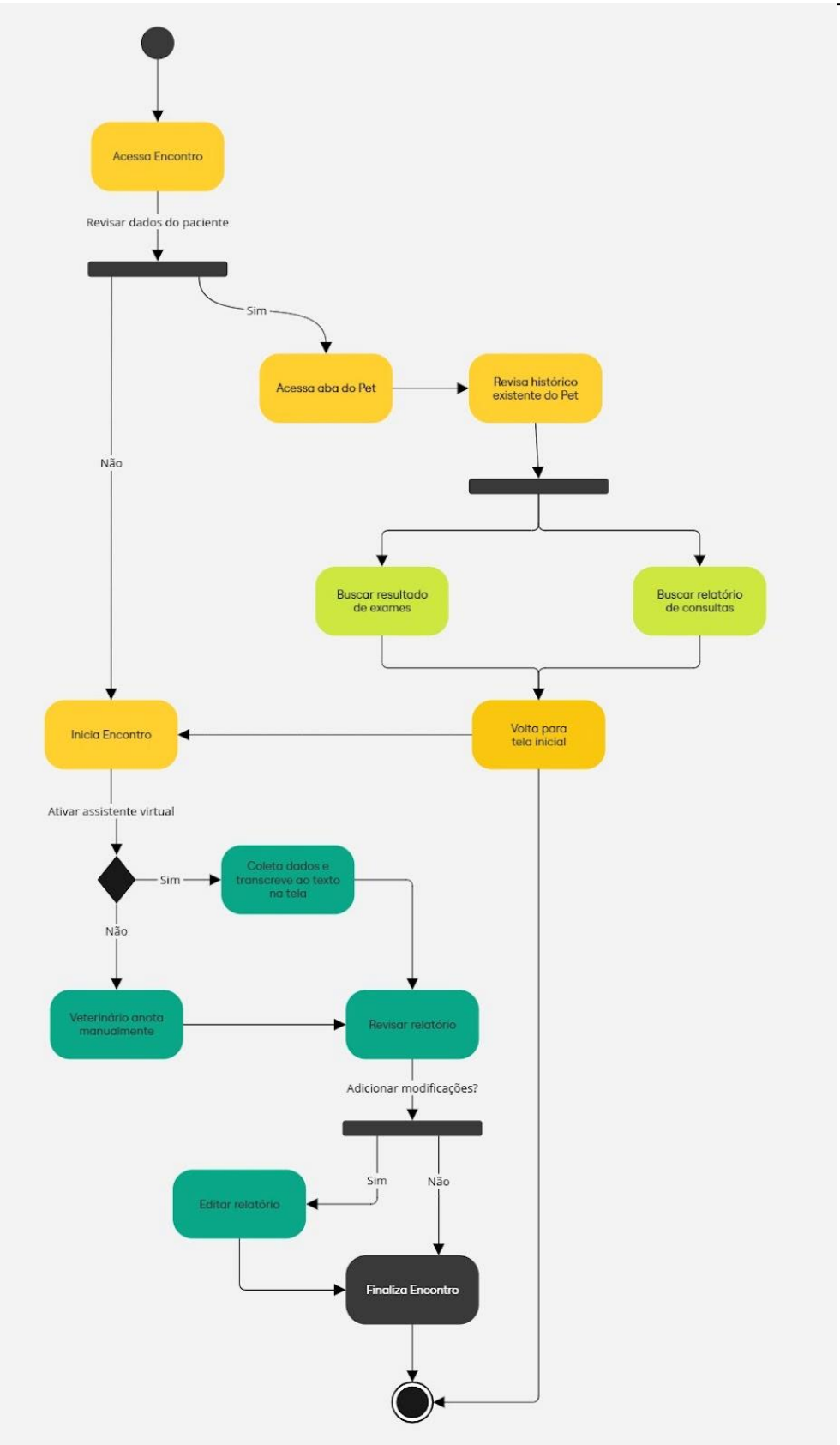
- **Tutor:** Nome, CPF, telefone, e-mail.
- **Pet:** Nome, espécie, idade, raça, tutor associado.
- **Encontro:** Data, hora, tipo (consulta, exame, cirurgia), status (pendente, confirmado).
- **Usuário:** Nome, e-mail, senha, tipo (administrativo, veterinário e usuário geral).

As relações entre essas classes, como associação entre tutores e pets ou entre encontros e veterinários, são detalhadas no diagrama.

4.4. Diagrama de Atividades

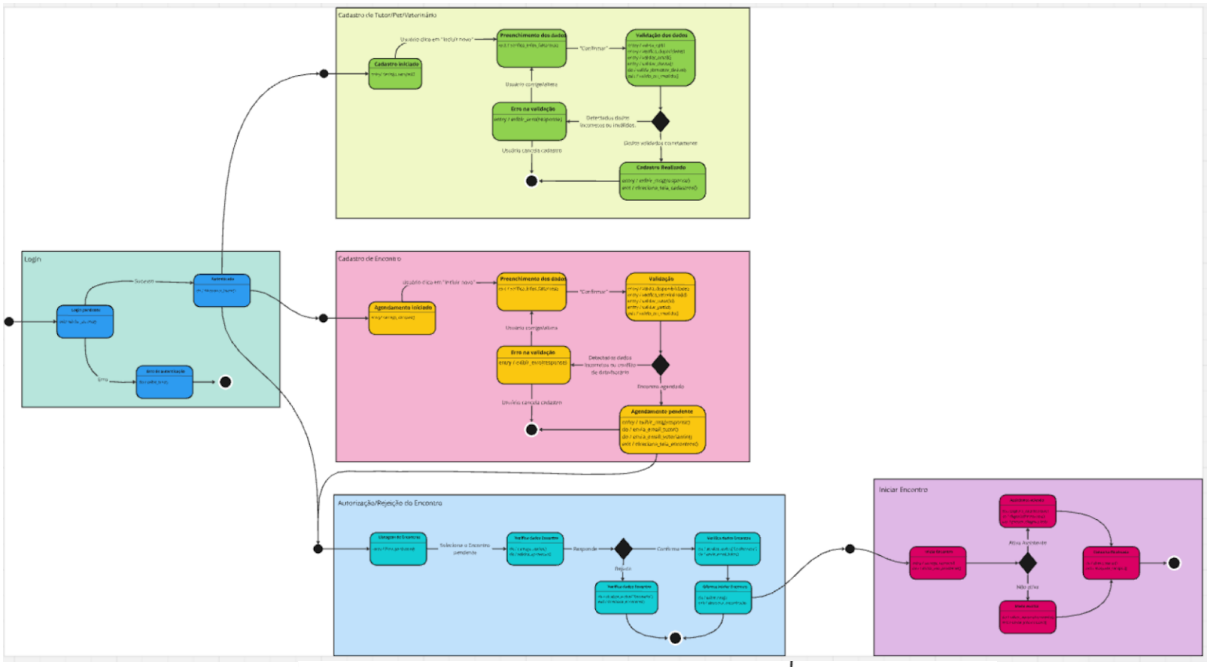
Para o diagrama de atividades implementei a partir da rotina “Realizar Consulta(Encontro)”, descrita abaixo:

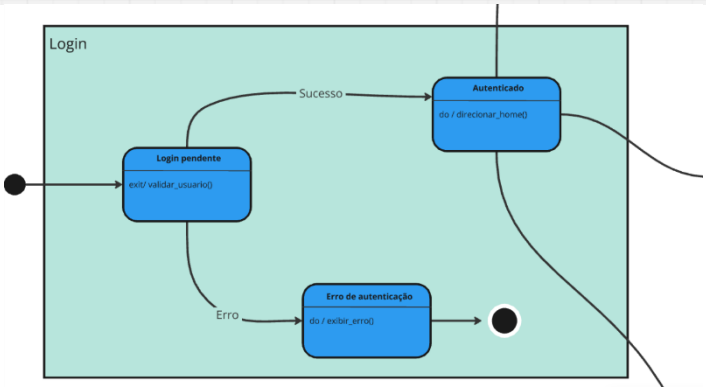
- Primeiro o usuário veterinário irá acessar o Encontro ao qual irá iniciar o atendimento
- Ao acessar, o usuário pode revisar as informações sobre o paciente ou apenas iniciar direto o Encontro sem revisar os dados do pet
- Caso o usuário decidir revisar os dados do paciente, ele poderá verificar os relatórios de exames e consultas passadas caso existirem, após isto o usuário retorna a tela inicial do Encontro
- Usuário irá clicar em iniciar o Encontro
- Será solicitado se desejará ativar o Assistente Virtual, caso não ativar o veterinário fará manualmente as anotações
- Caso ativado, o Assistente Virtual estará utilizando o microfone para captar entre as palavras ditas as informações relevantes para a consulta e anotará estas informações formatadas na tela
- O veterinário poderá fazer as próprias anotações manualmente de forma simultânea com o assistente
- Antes de finalizar o Encontro, o usuário poderá revisar o relatório sobre a consulta realizada e adicionar novas modificações
- O usuário finaliza o Encontro e o diagrama encerra

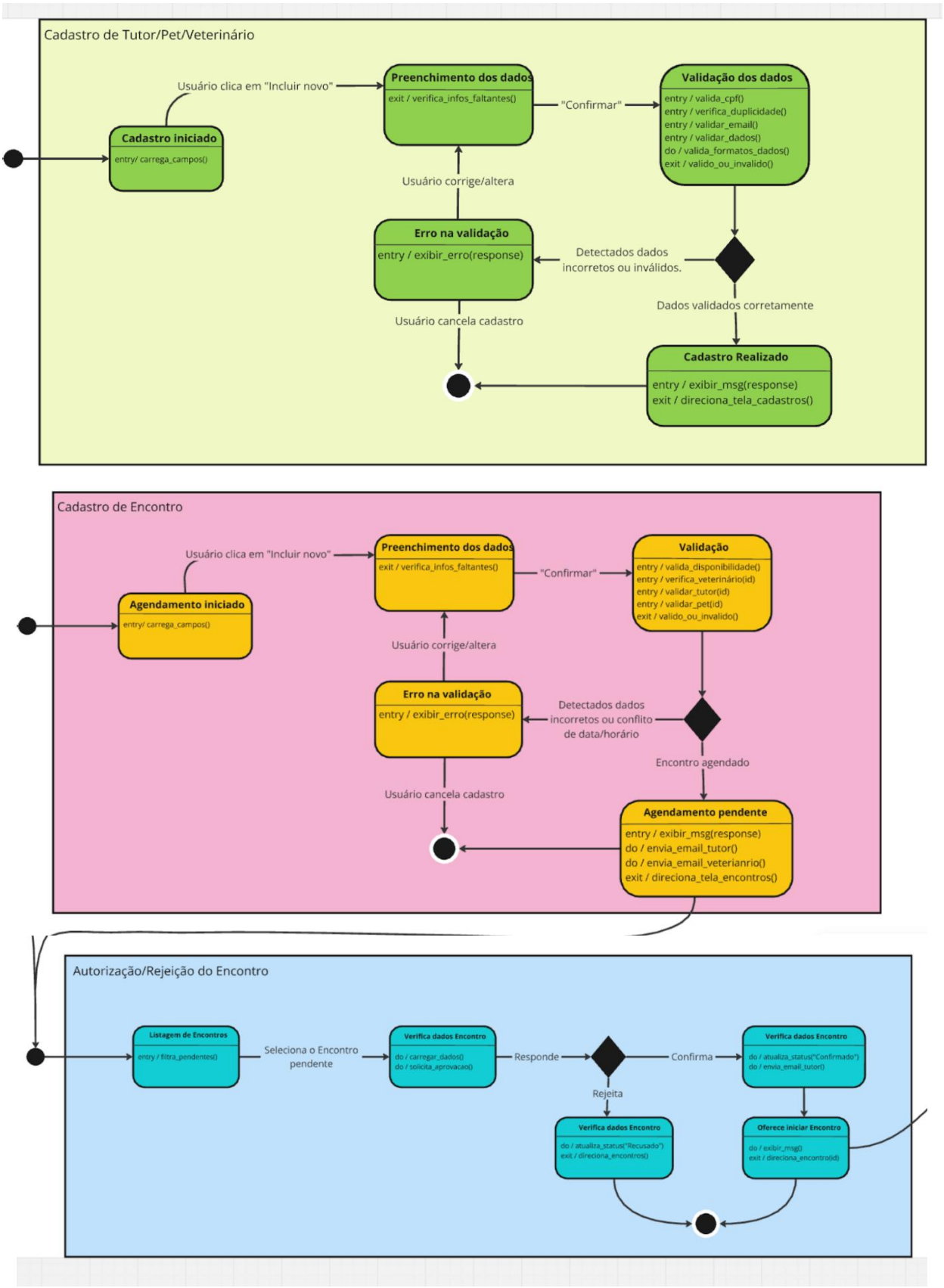


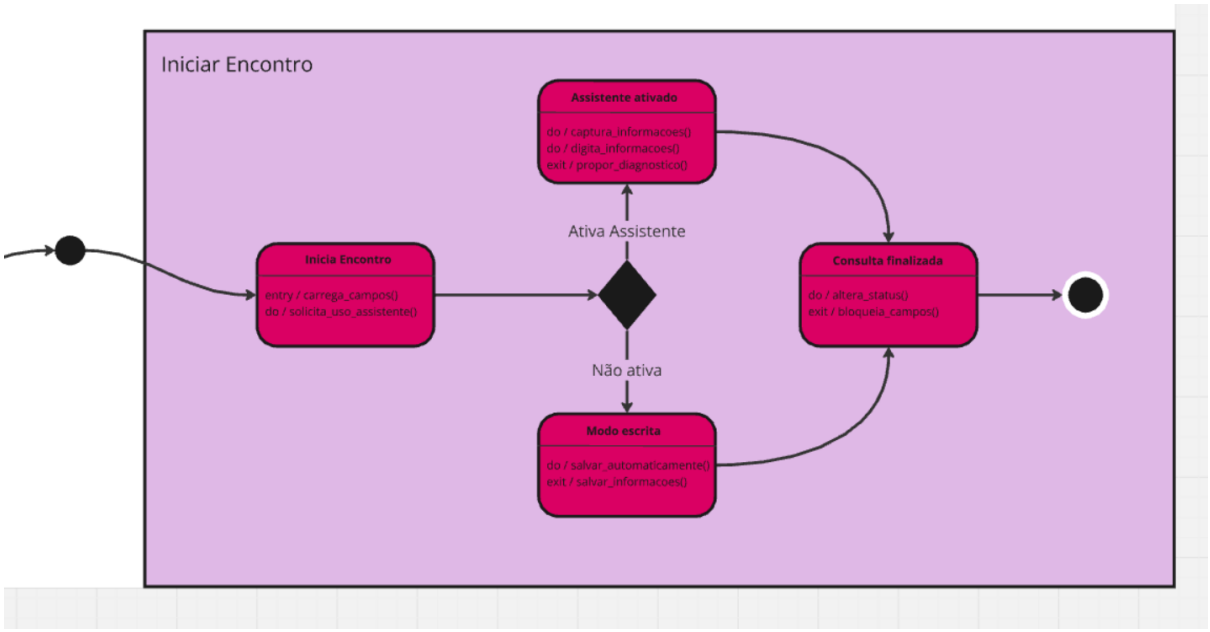
Notação do diagrama de atividades.

4.5. Diagrama de Estados



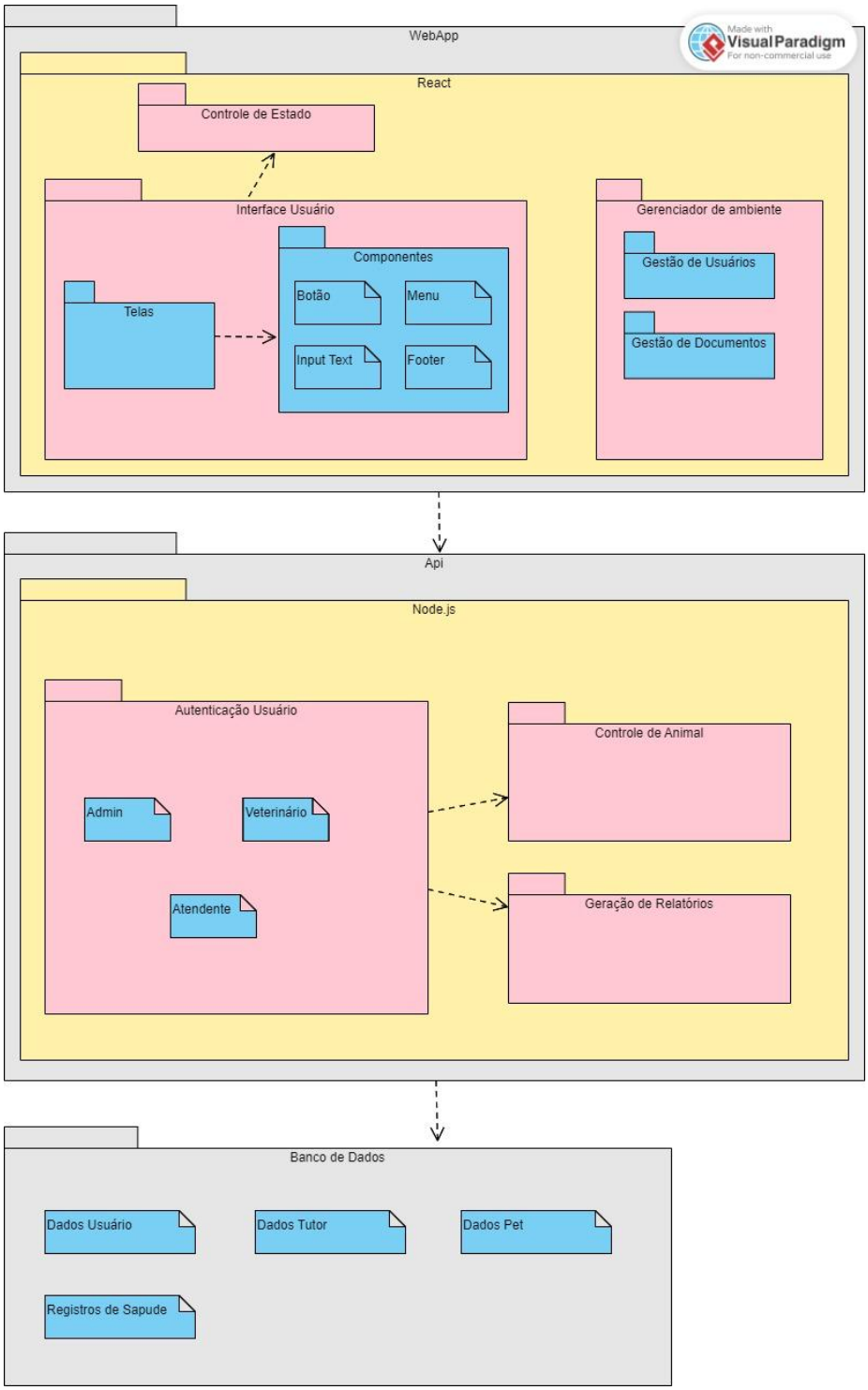



Cadastro de Tutor/Pet/Veterinário
Usuário clica em "Incluir novo"
Cadastro iniciado
entry/ carrega_campos()
Preenchimento dos dados
exit / verifica_infos_faltantes()
"Confirmar"
Validação dos dados
entry / valida_cpf()
entry / verifica_duplicidade()
entry / validar_email()
entry / validar_dados()
do / valida_formatos_dados()
exit / valido_ou_invalido()
Usuário corrige/altera
Erro na validação
entry / exibir_erro(response)
Detectados dados incorretos ou inválidos.
Dados validados corretamente
Usuário cancela cadastro
Cadastro Realizado
entry / exibir_msg(response)
exit / direciona_tela_cadastros()
Cadastro de Encontro
Usuário clica em "Incluir novo"
Agendamento iniciado
entry/ carrega_campos()
Preenchimento dos dados
exit / verifica_infos_faltantes()
"Confirmar"
Validação
entry / valida_disponibilidade()
entry / verifica_veterinário(id)
entry / validar_tutor(id)
entry / validar_pet(id)
exit / valido_ou_invalido()
Usuário corrige/altera
Erro na validação
entry / exibir_erro(response)
Detectados dados incorretos ou conflito de data/horário
Encontro agendado
Usuário cancela cadastro
Agendamento pendente
entry / exibir_msg(response)
do / envia_email_tutor()
do / envia_email_veterianrio()
exit / direciona_tela_encontros()
Autorização/Rejeição do Encontro
Listagem de Encontros
entry / filtra_pendentes()
Seleciona o Encontro pendente
Verifica dados Encontro
do / carregar_dados()
do / solicita_aprovacao()
Responde
Confirma
Verifica dados Encontro
do / atualiza_status("Confirmado")
do / envia_email_tutor()
Rejeita
Verifica dados Encontro
do / atualiza_status("Recusado")
exit / direciona_encontros()
Oferece iniciar Encontro
do / exibir_msg()
exit / direciona_encontro(id)



Notação do diagrama de estados.

4.6. Diagrama de Componentes



Notação do diagrama de componentes.

4.7. Modelo de Dados

4.8.1. Modelo Lógico da Base de Dados

Neste item deve ser apresentado o modelo lógico da base de dados, que pode ser o modelo entidade-relacionamento ou objeto da base de dados. No caso do modelo entidade-relacionamento o modelo lógico deve passar por todas as regras de normalização.

Como base para geração do modelo lógico pode-se utilizar o diagrama de classes. Geralmente ferramentas CASE geram automaticamente o modelo lógico da base de dados a partir do diagrama de classes.

4.8.2. Criação Física do Modelo de Dados

Neste item deve ser realizada a criação física do banco de dados, ou seja, a criação de *scripts*.

4.8.3. Dicionário de Dados

Neste item deve ser criado o dicionário de dados do banco de dados, com o obejtivo de documentar todas as tabelas, atributos, *stored procedures*[G].

4.8. Ambiente de Desenvolvimento

Neste item devem ser apresentados os softwares de desenvolvimento (linguagem de programação, banco de dados, ferramentas, etc.), equipamentos de hardware e redes que sejam essenciais para o desenvolvimento do sistema.

4.9. Sistemas e componentes externos utilizados

Neste item devem ser descritos os sistemas e componentes externos que serão utilizados no sistema. Por exemplo, sistemas que serão integrados ao sistema desenvolvido, componentes comprados ou livre que estão sendo utilizados para facilitar ou complementar o desenvolvimento do sistema.

5. Implementação

Este capítulo tem como objetivo a implementação das classes em termos de componentes, ou seja, A implementação do sistema VetTrack foi realizada com base nas definições estabelecidas nas fases de análise e design, utilizando os conceitos de programação orientada a objetos e boas práticas de desenvolvimento de software.

Linguagens e Tecnologias Utilizadas

- **Frontend**: React.js, com suporte a componentes reutilizáveis e estilização CSS modular.
- **Backend**: Node.js, utilizando o framework Express.js para criar APIs RESTful e aplicar as regras de negócio.
- **Banco de Dados**: PostgreSQL, com stored procedures e triggers para automação de processos críticos.
- **Assistente Virtual**: Implementado em Python, integrando um modelo de reconhecimento de voz baseado em bibliotecas como SpeechRecognition.

Estrutura de Implementação

Frontend

- Os componentes foram organizados em uma estrutura modular:
  - **Pasta components**: Contém os componentes reutilizáveis como formulários, botões e cabeçalhos.
  - **Pasta pages**: Abriga as páginas principais do sistema, como cadastro de pets, agendamento e relatórios.
  - **Pasta services**: Inclui chamadas para APIs externas.

Backend

- A arquitetura segue a separação de responsabilidades:
  - **Pasta routes**: Define as rotas da API, como /pets, /tutors, /appointments.
  - **Pasta controllers**: Contém a lógica para manipulação de dados recebidos nas requisições.
  - **Pasta models**: Representa as entidades do sistema (Tutor, Pet, Encontro) e suas interações com o banco de dados.
  - **Pasta middlewares**: Inclui autenticação JWT e validação de entrada.

Banco de Dados

- O banco foi estruturado com tabelas normalizadas para tutores, pets, encontros e usuários.
- Foram criadas stored procedures para operações comuns, como:
  - **sp_add_pet**: Insere um novo pet e associa a um tutor.
  - **sp_schedule_appointment**: Cria um novo encontro e atualiza a agenda do veterinário.

Boas Práticas Adotadas

1. **Cabeçalhos de Funções**: Cada função ou método implementado possui um cabeçalho contendo:
   a. Descrição: Propósito da função.
   b. Autor: Nome do desenvolvedor.
   c. Data de criação: Data inicial da implementação.
2. **Comentários no Código**: Foram adicionados comentários explicativos para cada bloco de lógica relevante.
3. **Padronização de Código**:
   a. Convenção de nomes: camelCase para variáveis e funções, PascalCase para classes.
   b. Organização dos arquivos por responsabilidade.
4. **Tratamento de Erros**:
   a. Utilização de blocos try/catch para capturar erros e retornar respostas consistentes para a interface.
   b. Logs detalhados para rastreamento de erros no backend.
5. **Padrões de Projeto Utilizados**:
   a. **Repository Pattern**: Para abstrair a camada de acesso ao banco de dados.
   b. **Singleton**: Garantir que o banco de dados tenha uma única instância ativa.
6. **Otimização do Código**:
   a. Consulta com índices no banco de dados para melhorar a performance de busca.
   b. Algoritmos eficientes para validações de dados e cálculos.

6. Testes

O processo de teste do sistema VetTrack foi planejado para identificar defeitos, validar a implementação dos requisitos e garantir a qualidade geral do software. Os testes cobriram funcionalidades essenciais, usabilidade, desempenho e segurança do sistema.

6.1. Plano de Testes

1. **Objetivos do Teste**:
   a. Garantir que os requisitos funcionais e não funcionais sejam atendidos.
   b. Verificar se o sistema é intuitivo e responsivo.
   c. Assegurar o desempenho sob cargas esperadas e a segurança dos dados.
2. **Tipos de Testes Realizados**:
   a. **Teste de Funcionalidade**: Validar operações como cadastro de pets, agendamento de encontros e geração de relatórios.
   b. **Teste de Usabilidade**: Avaliar a facilidade de uso da interface.
   c. **Teste de Desempenho**: Verificar a capacidade do sistema em processar múltiplas requisições simultâneas.
   d. **Teste de Segurança**: Garantir a proteção de dados sensíveis, como senhas e informações de usuários.
3. **Ferramentas Utilizadas**:
   a. **Frontend**: Cypress para testes end-to-end.
   b. **Backend**: Jest para testes unitários e de integração.
   c. **Banco de Dados**: pgTAP para testes em PostgreSQL.

6.2. Execução do Plano de Testes

1. Teste de Funcionalidade

Caso de Teste	Descrição	Resultado Esperado	Resultado Obtido
CT01 - Cadastro de Pet	Inserir dados válidos de um novo pet.	Pet cadastrado com sucesso.	Passou.
CT02 - Agendamento de Encontro	Agendar encontro com dados válidos.	Encontro registrado como pendente.	Passou.
CT03 - Cadastro de Usuário	Inserir e-mail inválido.	Exibir mensagem de erro.	Passou.

2. **Teste de Usabilidade**
   - **Critérios Avaliados**:
     - Navegação intuitiva.
     - Tempos de resposta das telas abaixo de 2 segundos.
     - Consistência visual entre dispositivo (desktop).
3. **Teste de Desempenho**
   - Cenário: Testar o sistema com 100 usuários simultâneos realizando operações de cadastro e consultas.
   - Ferramenta Utilizada: JMeter.
4. **Teste de Segurança**
   - **Cenário 1**: Testar a vulnerabilidade a ataques de injeção SQL.
   - **Cenário 2**: Testar a força da criptografia de senhas armazenadas.

7. Implantação

7.1. Manual de Implantação

1. **Configuração do Ambiente de Servidor**
    a. Instalação do Node.js para execução do backend.
    b. Configuração do PostgreSQL para armazenar os dados do sistema.
    c. Configuração de variáveis de ambiente (e.g., credenciais de banco de dados, chaves de API).
2. **Implementação do Backend**
    a. Deploy no servidor usando Docker para isolar e gerenciar os serviços.
    b. Utilização de um servidor web como NGINX para gerenciamento de requisições.
3. **Deploy do Frontend**
    a. Build do projeto React.js e upload para o serviço de hospedagem.
    b. Configuração de HTTPS usando certificados SSL para conexões seguras.
4. **Configuração de Rede e Segurança**
    a. Firewall configurado para bloquear acessos não autorizados.
    b. Definição de regras para acesso SSH e portas necessárias.
5. **Teste Pós-Implantação**
    a. Verificação do funcionamento de todas as funcionalidades principais (cadastro, agendamento, relatórios).
    b. Simulação de cenários de uso para garantir estabilidade.
6. **Treinamento do Usuário Final**
    a. Realização de sessões de treinamento com os funcionários administrativos e veterinários.
    b. Entrega do manual do usuário com instruções detalhadas.

8. Manual do Usuário

Este manual foi desenvolvido para auxiliar os usuários a utilizarem o sistema VetTrack de maneira eficiente. Ele apresenta instruções detalhadas sobre como realizar as principais ações no sistema, dividido por perfil de usuário.

Perfis de Usuário

1. **Veterinário**: Realiza consultas e registra diagnósticos e também pode gerenciar cadastros de tutores, pets e encontros.
2. **Funcionário**: Gerencia cadastros de tutores, pets e encontros.
3. **Administrador**: Gerencia usuários e configurações do sistema.

1. Acessando o Sistema

- **Passo 1**: Abra o navegador de sua preferência (Chrome recomendado) e acesse o endereço: 'https://vettrack.com'.
- **Passo 2**: Insira seu e-mail e senha no campo de login.
- **Passo 3**: Clique no botão **Entrar**.

Caso esqueça a senha, clique em **Esqueci minha senha** e siga as instruções enviadas por e-mail.

2. Cadastro de Tutores e Pets *(Perfil: Funcionário, veterinário)*

- **Passo 1**: No menu principal, clique em **Tutores**.
- **Passo 2**: Clique em **Adicionar Tutor, i**nsira as informações obrigatórias: nome, CPF, telefone e e-mail.
- **Passo 3**: Após cadastrar o tutor, clique em **Pets** no menu.
- **Passo 4**: Clique em **Adicionar Pet** e insira os dados: nome, espécie, idade e tutor associado.
- **Passo 5**: Clique em **Salvar**.

3. Agendamento de Encontros *(Perfil: Funcionário, veterinário)*

- **Passo 1**: No menu principal, selecione **Encontros**.
- **Passo 2**: Clique em **Agendar Encontro, e**scolha o pet e insira o motivo do encontro, tipo (consulta, exame ou cirurgia), data e hora.
- **Passo 3**: O sistema verificará a disponibilidade do veterinário.
- **Passo 4**: Clique em **Salvar** para confirmar o agendamento.

O tutor será automaticamente notificado por e-mail com os detalhes do encontro.

5. Geração de Relatórios *(Perfil: Todos os Usuários)*

- **Passo 1**: No menu principal, selecione **Relatórios**.
- **Passo 2**: Escolha os filtros desejados: período, tipo de encontro, status, etc.
- **Passo 3**: Clique em **Gerar Relatório**.
- **Passo 4**: O relatório será exibido e poderá ser baixado em formato PDF.

6. Gerenciamento de Usuários *(Perfil: Administrador)*

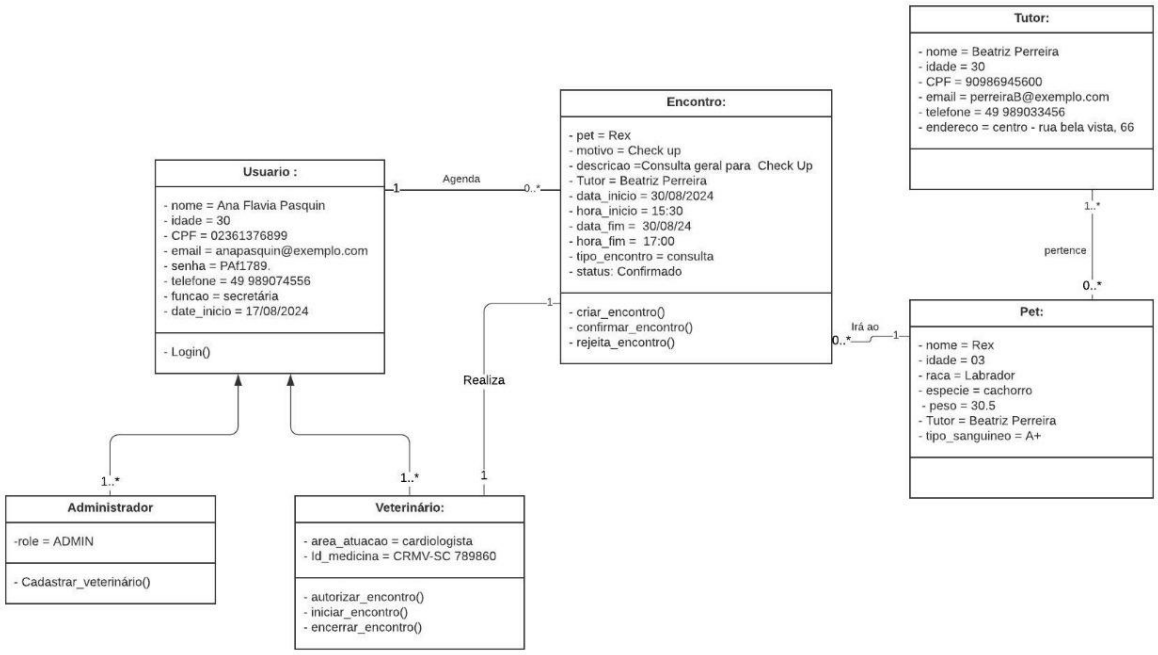
- **Passo 1**: Acesse o menu **Usuários**.
- **Passo 2**: Clique em **Adicionar Usuário**. Insira os dados do novo usuário, como nome, e-mail, senha e tipo de acesso.
- **Passo 3**: Clique em **Salvar** para concluir.

7. Boas Práticas

- Mantenha os dados sempre atualizados para evitar inconsistências.
- Faça logout ao finalizar o uso, especialmente em computadores compartilhados.

Este manual garante que todos os usuários consigam operar o sistema com facilidade. Caso encontre dificuldades, entre em contato com o suporte pelo e-mail: suporte@vettrack.com.

8.1. Diagrama de Objetos



9. Conclusões e Considerações Finais

O desenvolvimento do VetTrack demonstrou sua aplicabilidade como uma solução inovadora para o gerenciamento de consultas em clínicas veterinárias de pequeno e médio porte. O sistema foi projetado para atender às necessidades específicas do setor, garantindo organização, eficiência e praticidade no atendimento ao cliente.

Bibliografia

As seguintes fontes foram utilizadas para embasar o desenvolvimento do projeto VetTrack, abrangendo conceitos de engenharia de software, modelagem UML, desenvolvimento web e boas práticas de programação:

Livros e Artigos

- Pressman, Roger S. Engenharia de Software: Uma Abordagem Profissional. 8ª edição. McGraw Hill, 2016.
- Sommerville, Ian. Engenharia de Software. 10ª edição. Pearson, 2016.

Websites e Documentação Técnica

- React.js Documentation. Disponível em: https://reactjs.org/docs/
- Node.js Documentation. Disponível em: https://nodejs.org/en/docs/
- PostgreSQL Documentation. Disponível em: https://www.postgresql.org/docs/

Ferramentas e Softwares

- Lucidchart - Ferramenta para criação de diagramas UML. Disponível em: https://www.lucidchart.com/
- Miro - Ferramenta para criação de diagramas UML. Disponível em: https://miro.com

Glossário

Glossário

A

Administrador: Usuário responsável por gerenciar permissões, usuários do sistema e configurações gerais.

Agendamento: Processo de marcar um encontro (consulta, exame ou cirurgia) no sistema.

B

Banco de Dados: Repositório de informações estruturadas, utilizado para armazenar dados de tutores, pets, veterinários e encontros no sistema VetTrack.

C

Cadastro: Ato de registrar informações de um tutor, pet, usuário ou encontro no sistema.

Caso de Uso: Representação de uma interação específica entre um ator (usuário) e o sistema, descrita em diagramas UML.

Cliente: Dispositivo usado para acessar o sistema VetTrack, seja um computador ou smartphone.

Consulta: Tipo de encontro no qual o veterinário atende um pet para avaliação ou diagnóstico.

D

Diagrama de Classe: Representação gráfica das classes do sistema, seus atributos, métodos e relacionamentos.

Diagrama de Sequência: Representação das interações temporais entre objetos ou componentes do sistema.

E

Encontro: Termo usado no VetTrack para definir eventos agendados, como consultas, exames e cirurgias.

F

Frontend: Parte do sistema visível e interativa para o usuário, desenvolvida com React.js no VetTrack.

I

Interface: Tela ou conjunto de telas do sistema utilizadas para interação com o usuário.

Inteligência Artificial (IA): Tecnologia integrada ao VetTrack para auxiliar os veterinários com anotações automáticas e sugestões de diagnósticos.

M

Middleware: Componente do backend responsável por processar solicitações antes que sejam

atendidas pelo sistema principal, como autenticação de usuários.

P

Pet: Animal cadastrado no sistema, associado a um tutor.

PostgreSQL: Banco de dados relacional utilizado no VetTrack para armazenamento de informações.

R

Relatório: Documento gerado pelo sistema contendo informações organizadas, como histórico de encontros e dados de pets.

Requisito Funcional: Requisitos que descrevem o que o sistema deve realizar para atender às necessidades dos usuários.

Requisito Não Funcional: Requisitos que especificam restrições ou características de qualidade do sistema, como desempenho e segurança.

S

Segurança: Conjunto de práticas e mecanismos para proteger os dados sensíveis armazenados e processados pelo sistema.

Servidor: Máquina responsável por hospedar o backend e gerenciar as requisições enviadas pelos clientes.

T

Tutor: Pessoa responsável por um ou mais pets cadastrados no sistema.

U

Usuário: Qualquer pessoa que interage com o sistema VetTrack, como veterinários, funcionários administrativos e administradores.

V

Veterinário: Profissional que realiza consultas e procedimentos no sistema VetTrack, utilizando recursos como o assistente virtual.