**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA**

**ETEC DA ZONA LESTE**

**NOVOTEC Desenvolvimento de Sistemas**

**Miguel Gustavo de Sousa Campos**

**Pablo de Sousa Santos**

**Ricardo Luquetti Codo**

**Victor Hugo Navarro Taveira**

**CANISHERZ: Sistema embarcado para medição cardíaca em tempo real de cachorros domésticos**

**São Paulo**

**2024**

**Miguel Gustavo de Sousa Campos**

**Pablo de Sousa Santos**

**Ricardo Luquetti Codo**

**Victor Hugo Navarro Taveira**

**CANISHERZ: Sistema IoT para medição cardíaca em tempo real de cachorros domésticos**

Trabalho de Conclusão apresentado ao Curso do Ensino Médio com Habilitação Profissional de Técnico em Desenvolvimento de Sistemas da ETEC da Zona Leste, orientado pelo Prof. Jeferson Roberto de Lima, como requisito parcial para obtenção do título de técnico em Desenvolvimento de Sistemas.

**São Paulo**

**2024**

“A vida dos cães é muito curta. Sua única falha, realmente”.

**Agnes Sligh Turnbul**

**RESUMO**

A presença de cardiopatias em animais domésticos revela um problema recorrente, porém, de pouco conhecimento social. No Brasil, a grande maioria desses pets são cachorros, ressaltando a necessidade de uma solução eficiente. Este projeto propõe a composição de um equipamento vestível que monitora, em tempo real, a frequência cardíaca dos cães. Composto por um dispositivo e uma aplicação, o aparelho coletará os batimentos e alertará caso identifique irregularidades. Para o entendimento e aprofundamento do tema pautado, a metodologia quali-quantitativa mostra-se de grande relevância para a efetivação da pesquisa, entregando um volume satisfatório de dados comprovatórios, necessários para o molde e conclusão do projeto. A problemática evidenciada mostra-se um desafio para o animal e seus cuidadores. Seus sintomas, mesmo incomuns, são de difícil detecção, considerando a barreira comunicativa existente. Como resultado, almejamos a diminuição desses casos, além de auxiliar no atendimento, fornecendo uma melhor qualidade de vida para o cão. Em conclusão, espera-se que essa solução inovadora auxilie significativamente para a prevenção de emergências e a melhoria geral do bem-estar dos cães.

**Palavras-chaves:**Cardiopatias; Equipamento vestível; Aplicativo;Cães.

**ABSTRACT**

The presence of heart disease in domestic animals reveals a recurrent and little known problem in society. In Brazil, the vast majority of these animals are dogs and these cases highlight the need for an effective solution. This project proposes a wearable device that monitors the heart rate of dogs in real time. The device, consisting of a device and an application, will collect the heartbeat and alert in case of irregularities. To understand the problem, the case study methodology is relevant to the research, as it provides a satisfactory volume of data, necessary for the completion of the project. This problem is proving to be a challenge for the animal and its caretakers. Its symptoms, although infrequent, are difficult to detect given the existing communication barrier. Therefore, we aim to reduce the number of such cases, as well as to contribute to providing care and a better quality of life for the dog. In conclusion, this innovative solution is expected to significantly help prevent emergencies and improve the overall well-being of dogs.

**Keywords:** Heart disease; Wearable device; Application; Dogs.

**LISTA DE ILUSTRAÇÕES**

[Figura 1 – Exemplo Diagrama de caso de uso 12](#_Toc181469646)

[Figura 2 – Exemplo Diagrama de classe 13](#_Toc181469647)

[Figura 3 – Exemplo de diagrama de sequência 14](#_Toc181469648)

[Figura 4 – Exemplo de diagrama de atividade 15](#_Toc181469649)

[Figura 5 – Construção de um formulário em linguagem HTML 16](#_Toc181469650)

[Figura 6 – Exemplo formulário em HTML 17](#_Toc181469651)

[Figura 7 – Estilização do HTML utilizando o CSS 18](#_Toc181469652)

[Figura 8 – Estilização do CSS no HTML utilizado 19](#_Toc181469653)

[Figura 9 – A implementação do Javascript no HTML 20](#_Toc181469654)

[Figura 10 – Resultado do Javascript no HTML 21](#_Toc181469655)

[Figura 11 – Criação de projeto expo React Native 22](#_Toc181469656)

[Figura 12 – Exemplo de estrutura de pasta e um formulário básico em React Native 23](#_Toc181469657)

[Figura 13 – Continuação formulário básico, Estilização 25](#_Toc181469658)

[Figura 14 – Resultado do aplicativo de formulário 27](#_Toc181469659)

[Figura 15 – Resultado do aplicativo de formulário após pressionamento do botão 28](#_Toc181469660)

[Figura 16 – Esp32 29](#_Toc181469661)

[Figura 17 – Sensor MAX30102 30](#_Toc181469662)

[Figura 18 – Programa para manipulação de LEDs 31](#_Toc181469663)

[Figura 19 – Exemplo código C++ LED apagado 32](#_Toc181469664)

[Figura 20 – Exemplo código C++ LED aceso 33](#_Toc181469665)

[Figura 21 – Exemplo de Estrutura do Firecloud 34](#_Toc181469666)

[Figura 22 – Exemplo de Prototipação da Interface de Baixa Fidelidade 36](#_Toc181469667)

[Figura 23 – Exemplo de Prototipação da Interface de Alta Fidelidade 37](#_Toc181469668)

[Figura 24 – Diagrama de Caso de Uso CanisHerz 38](#_Toc181469669)

[Figura 25 – Diagrama de Atividade: Enviar Batimentos 41](#_Toc181469670)

[Figura 26 – Diagrama de Atividade: Cadastrar Tutor 41](#_Toc181469671)

[Figura 27 – Diagrama de Atividade: Cadastrar Cão 42](#_Toc181469672)

[Figura 28 – Diagrama de Atividade: Selecionar Cão 42](#_Toc181469673)

[Figura 29 – Diagrama de Atividade: Editar Cão 43](#_Toc181469674)

[Figura 30 – Diagrama de Atividade: Editar Tutor 43](#_Toc181469675)

[Figura 31 – Diagrama de Atividade: Excluir Cão 44](#_Toc181469676)

[Figura 32 – Diagrama de Atividade: Excluir Tutor 44](#_Toc181469677)

[Figura 33 – Diagrama de Atividade: Exibir Bateria 45](#_Toc181469678)

[Figura 34 – Diagrama de Atividade: Exibir Protocolo de Ajuda 45](#_Toc181469679)

[Figura 35 – Diagrama de Atividade: Gerenciar Bateria 46](#_Toc181469680)

[Figura 36 – Diagrama de Atividade: Gerenciar Batimentos 46](#_Toc181469681)

[Figura 37 – Diagrama de Atividade: Realizar Login 47](#_Toc181469682)

[Figura 38 – Diagrama de Atividade: Selecionar Cão 47](#_Toc181469683)

[Figura 39 – Diagrama de Atividade: Visualizar Cão 48](#_Toc181469684)

[Figura 40 – Diagrama de Atividade: Visualizar Tutor 48](#_Toc181469685)

[Figura 41 – Diagrama de Atividade: Acompanhar Batimentos 48](#_Toc181469686)

[Figura 42 – Diagrama de Sequência: Acompanhar Batimentos 49](#_Toc181469687)

[Figura 43 – Diagrama de Sequência: Cadastrar Cachorro 49](#_Toc181469688)

[Figura 44 – Diagrama de Sequência: Cadastrar Tutor 50](#_Toc181469689)

[Figura 45 – Diagrama de Sequência: Editar Cachorro 50](#_Toc181469690)

[Figura 46 – Diagrama de Sequência: Editar Tutor 51](#_Toc181469691)

[Figura 47 – Diagrama de Sequência: Enviar Batimentos 51](#_Toc181469692)

[Figura 48 – Diagrama de Sequência: Excluir Cachorro 52](#_Toc181469693)

[Figura 49 – Diagrama de Sequência: Excluir Tutor 52](#_Toc181469694)

[Figura 50 – Diagrama de Sequência: Excluir Tutor 53](#_Toc181469695)

[Figura 51 – Diagrama de Sequência: Listar Cachorros 53](#_Toc181469696)

[Figura 52 – Diagrama de Sequência: Realizar Login 54](#_Toc181469697)

[Figura 53 – Diagrama de Máquina de estado: Autenticar Tutor 54](#_Toc181469698)

[Figura 54 – Diagrama de Máquina de estado: Acompanhar Batimentos 55](#_Toc181469699)

[Figura 55 – Diagrama de Máquina de estado: Cadastrar Tutor/Cão 56](#_Toc181469700)

[Figura 56 – Diagrama de Máquina de estado: Editar Tutor/Cão 56](#_Toc181469701)

[Figura 57 – Diagrama de Máquina de estado: Excluir Tutor/Cão 56](#_Toc181469702)

[Figura 58 – Diagrama de Máquina de estado: Exibir Bateria 57](#_Toc181469703)

[Figura 59 – Diagrama de Máquina de estado: Selecionar Cão 57](#_Toc181469704)

[Figura 60 – Esquemática de componentes Iot CanisHerz 58](#_Toc181469705)

[Figura 61 – Prototipação de tela: Boas-Vindas 59](#_Toc181469706)

[Figura 62 – Prototipação de tela: Entrar 60](#_Toc181469707)

[Figura 63 – Prototipação de tela: Cadastrar Tutor 60](#_Toc181469708)

[Figura 64 – Prototipação de tela: Cadastrar Cachorro 61](#_Toc181469709)

[Figura 65 – Prototipação de tela: Proposito 61](#_Toc181469710)

[Figura 66 – Prototipação de tela: Carregamento 62](#_Toc181469711)

[Figura 67 – Prototipação de tela: Inicio (Sem conexão) 62](#_Toc181469712)

[Figura 68 – Prototipação de tela: Inicio (Com conexão) 63](#_Toc181469713)

[Figura 69 – Prototipação de tela: Medidor 63](#_Toc181469714)

[Figura 70 – Prototipação de tela: Medidor (Instruções fora da média) 64](#_Toc181469715)

[Figura 71 – Prototipação de tela: Conta 64](#_Toc181469716)

[Figura 72 – Prototipação de tela: Novo Cachorro 65](#_Toc181469717)

[Figura 73 – Prototipação de tela: Editar Informações 65](#_Toc181469718)

**LISTA DE QUADROS**

**LISTA DE ABREVIATURAS E SIGLAS**

Cascading Style Sheet (CSS)

Hypertext Markup Language (HTML)

integrated development environment (IDE)

Internet of Things (IoT)

JavaScript (JS)

Light Emitting Diode (LED)

Npx (Node Package Executor)

Structured Query Language (SQL)

Computer Aided Design (CAD)

Voltage Common Collector (VCC)

Voltage Input (VIN)

Ground (GND)

Serial Clock Line (SCL)

Serial Data Line (DAS)

General Purpose Input/Output (GPIO)

[1 INTRODUÇÂO 10](#_Toc181469612)

[2 REFERENCIAL TÉORICO 11](#_Toc181469613)

[2.1 Cardiopatias 11](#_Toc181469614)

[2.2 Uml 11](#_Toc181469615)

[2.3 Html 15](#_Toc181469616)

[2.3.1 Funcionamento do HTML na prática 15](#_Toc181469617)

[2.4 Css 17](#_Toc181469618)

[2.5 Javascript 19](#_Toc181469619)

[2.6 React 21](#_Toc181469620)

[2.7 React Native 21](#_Toc181469621)

[2.7.1 Expo 22](#_Toc181469622)

[2.7.2 Criação de aplicação em React Native 22](#_Toc181469623)

[2.8 IoT 29](#_Toc181469624)

[2.9 ESP32 29](#_Toc181469625)

[2.10 Sensor MAX30102 30](#_Toc181469626)

[2.11 Battery Shield V8 30](#_Toc181469627)

[2.12 C 30](#_Toc181469628)

[2.13 C++ 30](#_Toc181469629)

[2.17 Banco de dados 33](#_Toc181469630)

[2.18 Firebase 34](#_Toc181469631)

[2.18.1 Firestore 34](#_Toc181469632)

[2.19 Impressão 3D 35](#_Toc181469633)

[2.20 Modelagem 3D 35](#_Toc181469634)

[2.21 Prototipação 36](#_Toc181469635)

[3 DESENVOLVIMENTO 38](#_Toc181469636)

[3.1 Diagrama de Caso de Uso 38](#_Toc181469637)

[3.2 Documentação dos Casos de Uso 39](#_Toc181469638)

[3.3 Diagrama de atividade 40](#_Toc181469639)

[3.4 Diagrama de Sequência 49](#_Toc181469640)

[3.5 Diagrama de Máquina de Estado 54](#_Toc181469641)

[3.6 Modelagem do sistema IoT 57](#_Toc181469642)

[3.7 Prototipação de telas da aplicação mobile 59](#_Toc181469643)

[4 Conclusão 66](#_Toc181469644)

[REFERENCIAS 68](#_Toc181469645)

1 INTRODUÇÂO

A presença de cardiopatias em animais domésticos revela-se um problema recorrente, porém, de pouco conhecimento social. Segundo PETMAG (2019), o Brasil possui aproximadamente 57 milhões de animais de estimação, onde 7% das mortes desses bichos são causadas por cardiopatias. Nessa porcentagem, 37 milhões são compostos por cachorros, sendo maioria nos casos contabilizados. As causas do desenvolvimento dessas doenças são variadas, como aponta o CONSELHO REGIONAL DE MEDICINA VETERINÁRIA DO ESTADO DE SÃO PAULO (2019), podendo ser o resultado do sedentarismo, obesidade, idade avançada, alimentação e outras circunstâncias que favorecem a aparição de sintomas relacionados. Conforme Fraga (2014), os principais sinais da presença de complicações cardiovasculares envolvem tosse, perda ou ganho de peso, dificuldade respiratória, edemas e outras complicações que exigem uma maior atenção caso identificados.

Pensando na melhoria do atendimento e detecção de doenças existentes, este projeto estrutura-se a fim de auxiliar no monitoramento de cães que se enquadram nas características citadas, permitindo um melhor entendimento de suas condições médicas. Para isso, é proposto a criação de uma ferramenta que permitirá a detecção da frequência cardíaca do animal, por meio de uma vestimenta portadora de um sensor de batimentos, responsável pela coleta e armazenamento desses dados. Como parte do sistema, será desenvolvido uma aplicação para exibição e gerenciamento das informações, fornecendo ao tutor do cachorro todas as informações necessárias para o entendimento da saúde do cão. Esse aplicativo, emitirá alarmes e notificações em situações de alteração drástica na frequência cardíaca, como forma de agilizar o atendimento em casos de emergência e minimizar as ocorrências de fatalidades decorrentes a cardiopatias.

2 REFERENCIAL TÉORICO

Essa seção abrange a documentação teórica que fundamenta o projeto, atribuindo noções fundamentais sobre as tecnologias que serão empregadas na criação e elaboração do dispositivo CanisHerz.

2.1 Cardiopatias

Conforme Feldman (2024), A cardiopatia é um termo geral que se refere a qualquer doença do coração. Isso pode incluir uma variedade de condições, como arritmias cardíacas.

Franco (2022), afirma que os efeitos negativos das doenças cardíacas em pacientes veterinários podem afetar o tempo de sobrevivência dos animais, visando analisar a qualidade de vida de cães com cardiopatias.

O sistema muscular responsável pela excitação e condução controla a função cardíaca. Segundo Jericó, Neto e Kogika (2015), O ritmo e a eficácia da bomba cardíaca são diretamente afetados pelo sistema de condução.

2.2 Uml

UML é uma linguagem para modelar softwares, adotada na indústria de engenharia de software, de propósito geral aplicada a todos os domínios. Sendo uma ferramenta padrão globalmente reconhecida como demonstrado por Guedes (2018).

Sua criação foi um processo experimental diz Fowler et al (2005). O resultado é um padrão baseado em diversas ideias e contribuições feitas por diversos colaboradores durante o processo.

Mencionado por Booch, Rumbaugh e Jacobson (2012), ela possui uma semântica precisa, garantindo interpretação clara por qualquer desenvolvedor ou ferramenta. Isso assegura que os modelos possam ser compreendidos sem ambiguidades.

A UML é composta por vários diagramas, com o objetivo de expor uma variedade de visões do sistema, em vários aspectos, dessa maneira, almejando uma modelagem com um número mínimo de falhas segundo Guedes (2018).

Entre esses vários tipos de diagramas, uma parte é utilizado nesse projeto, como: Diagrama de Caso de Uso, Diagrama de Classe, Diagrama de Atividade e Diagrama de Sequência.

O diagrama de caso de uso fornece a visão das funcionalidades para os usuários. Concentra-se na representação das interações do sistema, sem entrar em detalhes de implementação, como dito por Booch, Rumbaugh e Jacobson (2012).

Figura 1 – Exemplo Diagrama de caso de uso

Diagrama

Descrição gerada automaticamente

Fonte: Do próprio autor, 2024

A imagem acima mostra as responsabilidades de um Atendente e um Gerente em uma loja de sapatos. O Atendente pode vender sapatos e confirmar pedidos, que pode incluir a nota fiscal e opcionalmente adicionar o cpf. O Gerente tem a tarefa de manter estoque, porém herda todas as funções do atendente, podendo realizar vendas de sapatos. Ambos os atores requerem Fazer Login para suas tarefas.

Conforme Fowler et al (2005), o diagrama de classes descreve os tipos de objetos e seus relacionamentos estáticos no sistema, incluindo propriedades, operações e restrições. Eles oferecem uma visão estruturada das classes e suas características.

Figura 2 – Exemplo Diagrama de classe

Diagrama

Descrição gerada automaticamente

Fonte: Do próprio autor, 2024

Acima é apresenta duas classes, cliente e compra. Cliente possui atributos como id, nome, email, senha e endereço, e métodos para login e registro do comprador. Compra tem atributos para o valor do produto e a data da compra, assim como um método para registrar compras realizadas. E temos uma relação “realiza” entre Cliente e Compra, indicando que um cliente pode realizar compras.

Parafraseando Guedes (2018), o diagrama de sequência tem o objetivo de detalhar a ordem precisa de eventos, mensagens enviadas e métodos chamados, oferecendo uma visão clara de como os objetos interagem durante os processos.

Figura 3 – Exemplo de diagrama de sequência

Diagrama

Descrição gerada automaticamente

Fonte: Do próprio autor, 2024

O diagrama de sequência "Checar Estoque" apresentado anteriormente demonstra o procedimento de verificação de disponibilidade de um modelo de sapato no estoque de uma loja. O Gerente começa o processo na interface gráfica "Visão Checar Estoque", selecionando o modelo que deseja ver a quantidade disponível no estoque. A escolha é enviada ao "Controle Checar Estoque", que consulta o estoque. O estoque responde com a informação, retornando-a à interface gráfica para visualização do gerente. As lifelines mostram a existência dos objetos durante a interação, enquanto as setas preenchidas indicam mensagens e as pontilhadas representam retornos de informação.

O diagrama de atividades como relembrado por Booch, Rumbaugh e Jacobson (2012), apresenta como uma atividade leva a outra em um processo. Essas atividades resultam em ações concretas, como mudanças de estado no sistema.

Figura 4 – Exemplo de diagrama de atividade

Diagrama

Descrição gerada automaticamente

Fonte: Do próprio autor, 2024

O diagrama apresenta o processo de cadastro, começando com a ação do usuário para iniciar o cadastro. Ele preenche as informações necessárias. O sistema confirma se os dados apresentados são corretos. Caso corretos, o cadastro é concluído com sucesso, se não, o usuário é solicitado a reinserir os dados.

O diagrama de máquina de estados apresenta o comportamento de um elemento por meio de um grupo finito de transições de estado. Além de ser utilizado para expressar o comportamento de uma parte do sistema como lembrado por Fowler et al (2005).

2.3 Html

A linguagem HTML, traduzida para o português como linguagem de marcação de hipertexto citado pelo autor Silva (2008), tem o objetivo de criar sites de computadores com uma estrutura de visualização mais simples.

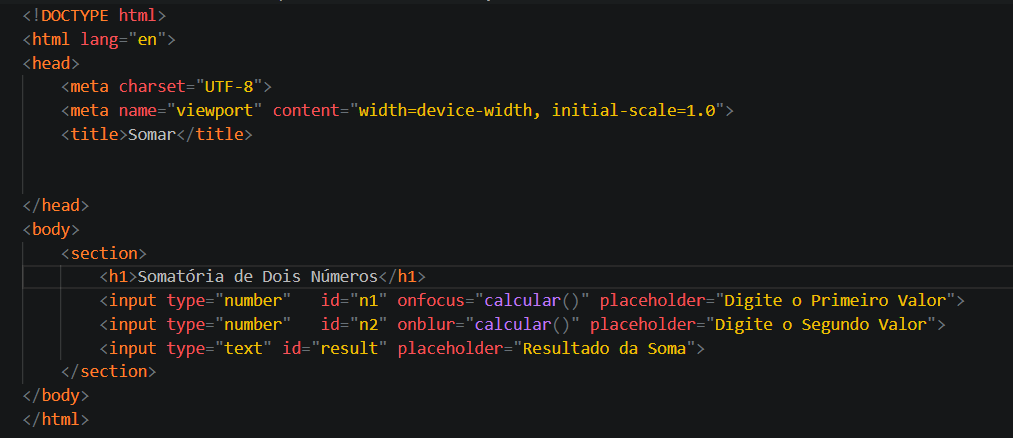
Segundo o Flatschart (2011) destaca-se que a linguagem de marcação de hipertexto, pode ser criada de forma semântica, ou seja, permitindo a maneira que será projetada na janela de visualização da página.

Portanto, o acesso a esses códigos é feito a via Browser, utilizando programas como o *Google,* *Internet Explorer,* entre outros tipos de *softwares,* conforme referidopor Marly (2012).

2.3.1 Funcionamento do HTML na prática

Usaremos um bloco de notas ou uma ferramenta específica para programação, para criar um tipo de arquivo HTML e estabelecer o escopo do código desta linguagem.

Figura 5 – Construção de um formulário em linguagem HTML



Fonte: Do próprio autor, 2024

Esse exemplo, mostra um formulário simples na estrutura da linguagem abordada anteriormente, utilizando os seguintes blocos de comandos:

* <!DOCTYPE html>: Declara o tipo de versão do HTML.
* <html lang=”en”>: Determina o idioma da estrutura HTML que será abordada.
* <head>: Inicia as informações dos registros que não são expostos.
* <meta charset=”UTF-8”>: Específica a codificação de caracteres para que o navegador execute corretamente os caracteres internacionais e especiais.
* <meta name=”viewport” content=”width=device-width, initial-scale=1.0”>: São configurações do *viweport* (janela de exibição) para dimensões de dispositivos móveis.
* <title>: Declara o nome dá página, que no exemplo utilizado é “Somar”.
* <body>: É responsável pelo conteúdo visível da documentação.
* <section>: Define uma seção no documento, que neste caso contém o formulário para realizar a soma de dois números.
* <h1>: Denomina o título de nível 1 de espessura.
* <input type="number" id="n1" onfocus="calcular()" placeholder="Digite o Primeiro Valor">: Esta *tag* é um campo de digitação, utilizando o atributo chamado *'type'* para especificar o tipo de caracteres que será utilizado. O atributo *'placeholder'* também pode ser utilizado para fornecer um texto de exemplo dentro do campo de entrada. Neste caso, são apenas números.

Figura 6 – Exemplo formulário em HTML



Fonte: Do próprio autor, 2024

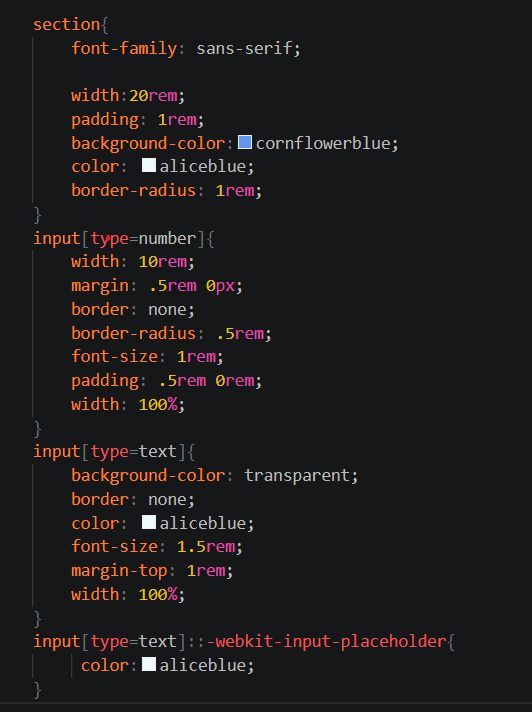
2.4 Css

Envidenciado pelo [Scheidt](https://www.google.com.br/search?hl=pt-BR&tbo=p&tbm=bks&q=inauthor:%22Felippe+Alex+Scheidt%22) (2014), CSS (Cascading Style Sheets) ou Folhas de Estilo em Cascata é uma linguagem de estilo para a linguagem de marcação de hipertexto.

Segundo Miletto e Bertagnolli (2015) o CSS propõe a possibilidade de criar estilos personalizados para títulos, listas, botões, entre outros elementos visuais da documentação do HTML.

Esta linguagem pode ser programada em qualquer editor de HTML ou Bloco de notas. Sendo acessível, o CSS é bastante utilizado em projetos que envolve a linguagem de marcação de elementos, citado por Jobstraibizer (2009).

Figura 7 – Estilização do HTML utilizando o CSS

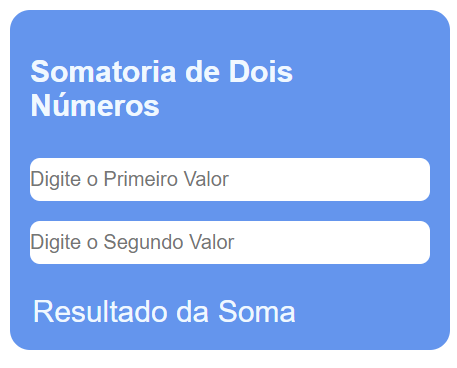


Fonte: Autoria Própria

Estes são os comandos de estilização do HTML feito em blocos de comando, visando as marcações do HTML:

* section {...}: Este bloco de código aplica estilos a elementos HTML dentro de uma seção. Ele define a fonte como "sans-serif", define a largura como 20rem, adiciona um preenchimento interno de 1rem, define a cor de fundo como "cornflowerblue", a cor do texto como "aliceblue" e aplica um raio de borda de 1rem.
* input[type=number] {...}: Este bloco de código aplica estilos a elementos de entrada do tipo "number". Ele define a largura como 10rem, adiciona margens de 0.5rem acima e abaixo, remove a borda, define um raio de borda de 0,5rem, define o tamanho da fonte como 1rem, adiciona um preenchimento de 0,5rem vertical e define a largura como 100%.
* input[type=text] {...}: Este bloco de código aplica estilos a elementos de entrada do tipo "text". Ele define a cor de fundo como transparente, remove a borda, define a cor do texto como "aliceblue", define o tamanho da fonte como 1.5rem, adiciona uma margem superior de 1rem e define a largura como 100%.
* input[type=text]::-webkit-input-placeholder {...}: Este bloco de código aplica estilos ao espaço reservado (placeholder). Ele define a cor do espaço reservado como "aliceblue", garantindo que o texto de *placeholder* seja de outra cor.

Figura 8 – Estilização do CSS no HTML utilizado



Fonte: Autoria Própria

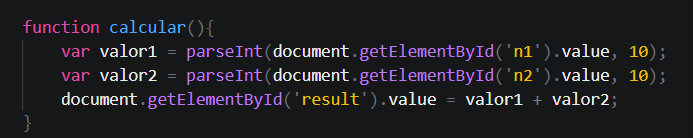
2.5 Javascript

Javascript é uma linguagem dinâmica e interpretada, que segundo Flanagan (2013) não requer tipagem, tornando-a adequada para abordagens orientadas a objetos e funcionais.

Parafraseando Morrison (2008) ela é a chave para transformar um projeto web em uma página interativa, capacitando os elementos que podem responder às suas ações, processar entradas do usuário e atender às suas necessidades.

A linguagem não possui conceito de classe como apresentado por Zakas (2017), em vez disso, utiliza dois tipos: os primitivos que são armazenados como tipos simples e os tipos de referência que são armazenados como objetos.

Figura 9 – A implementação do Javascript no HTML

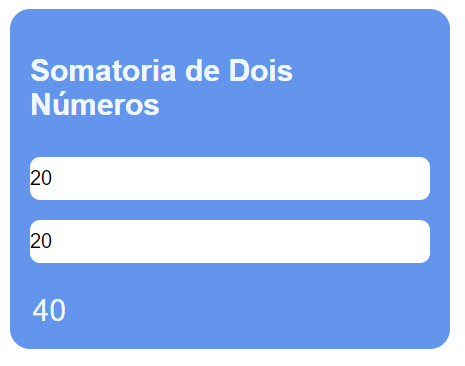


Fonte: Do próprio autor, 2024

Nesse é exemplo, são os comandos para o funcionamento “somar” do formulário criado por HTML e estilizado por CSS:

* Function calcular() {...}: Isso define uma função chamada "calcular". Neste caso, a função "calcular" será chamada quando desejarmos realizar cálculo da soma.
* Var valor1 = parseInt(document.getElementById('n1').value, 10): Esta linha declara uma variável chamada "valor1" e utiliza a função *parseInt* para converter o valor obtido do elemento HTML com o ID 'n1' para um número inteiro. O método *getElementById*('n1') obtém o elemento HTML com o ID 'n1', e o ".*value*" recupera o valor inserido nesse elemento. O *parseInt* converte esse valor para um número inteiro.
* Var valor2 = parseInt(document.getElementById('n2').value, 10): Similarmente à linha anterior, esta declara uma variável chamada "valor2" e converte o valor do elemento HTML com o ID 'n2' para um número inteiro.
* Document.getElementById('result').value = valor1 + valor2: Esta linha define o valor do elemento HTML com o ID 'result' como a soma dos valores de "valor1" e "valor2". O método getElementById('result') obtém o elemento HTML com o ID 'result', e ".value" define o valor do elemento. O valor atribuído é a soma dos valores de "valor1" e "valor2", realizada pela operação de adição (+).

Figura 10 – Resultado do Javascript no HTML



Fonte: Do próprio autor, 2024

2.6 React

O React é uma biblioteca JavaScript capaz de facilitar o desenvolvimento de interfaces gráficas voltadas para o âmbito *web*, agilizando o processo de criação dessas telas e disponibilizando uma maior interação do usuário ao sistema (React, 2024).

Segundo Silva (2021), essa ferramenta, criada pelo Facebook, mostra-se uma das mais populares no ramo, sendo utilizada por grandes empresas como a Netflix, PayPal e Twitter, evidenciando sua eficácia no que propõe.

2.7 React Native

Como explicado por Escudelario e Pinho (2020), O React Native é considerado uma biblioteca do React, que disponibiliza várias ferramentas que auxiliam na criação de aplicações moveis, para iOS e Android, utilizando linguagens de suporte como HTML e CSS.

Foi introduzido em 2015 pelo Facebook. Permitindo aos desenvolvedores reutilizar conhecimentos das tecnologias web para o desenvolvimento de aplicativos móveis, facilitando a criação dos mesmos mencionado por Bezerra e Viana (2021).

Ele adota uma sintaxe do ReactJS, pelo uso do JSX. Porém, sua abordagem difere de sua raiz. Devido os seus elementos serem emulados de forma nativa, utilizando o JS Core como uma ponte entre os dois como citado por Silva e Sousa (2019).

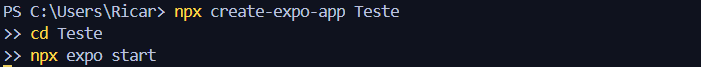
2.7.1 Expo

Segundo Alura (2022), a ferramenta expo, é um conjunto de ferramentas e serviços em torno de plataformas nativas que ajudam no desenvolvimento, construção, implementação e permitem a criação rapidamente de aplicativos iOS, Android e web.

2.7.2 Criação de aplicação em React Native

Utilizando a ferramente anteriormente mencionada, e disposto os seguintes comandos no *Prompt* de Comando do Windows:

Figura 11 – Criação de projeto expo React Native



Fonte: Do próprio autor, 2024

Os comandos acima utilizando o prefixo npx que permite executar comandos de pacotes diretamente, sem precisar instalá-los globalmente ou localmente, respectivamente farão:

* A criação do ambiente do projeto, com o nome informado;
* Direcionamento de diretório do Prompt para a pasta informada;
* Inicialização da aplicação no Expo, permitindo uma visualização mediante ao aplicativo Expo Go, instalado em um dispositivo móvel.

Após a execução, no diretório será criado uma variedade de pastas para configuração do ambiente da aplicação, assim o projeto poderá ser desenvolvido em uma IDE de preferência, modificando incialmente o arquivo app.js para alteração das *views*.

Figura 12 – Exemplo de estrutura de pasta e um formulário básico em React Native

Texto

Descrição gerada automaticamente Fonte: Do próprio autor, 2024

Nesse exemplo é apresentado um código de formulário simples como exemplo de aplicação, utilizando os seguintes elementos:

* import { View, Text, TextInput, Button, Alert, StyleSheet } from 'react-native': É utilizado para importar seis componentes do pacote padrão do react, fundamentais para construir interfaces e utilizar os comandos seguintes.
* import React, { useState } from 'react': Essa linha importa o React e a função useState. Ela é uma função especial que permite adicionar estado aos componentes.
* const [name, setName] = useState(''); const [email, setEmail] = useState(''); const [message, setMessage] = useState(''): Esses comandos estão criando três estados: *name*, *email* e *message*, que serão usados para armazenar os valores do formulário. Inicializando cada estado com uma *string* vazia.
* const handleSubmit = () => É a criação de uma função *handleSubmit*, que será chamada quando o botão "Enviar" for apertado.
* Alert.alert('Dados do Formulário', `Nome: ${name}\nEmail: ${email}\nMensagem: ${message}`): É utilizado para exibir um alerta com os dados inseridos no formulário posteriormente.
* <View>: Essa *tag* abre uma View. Essa View serve como um recipiente para os elementos do formulário.
* <Text>: Usada para representar texto em linha exibindo o mesmo.
* <TextInput>: Permite que os usuários insiram textos. Pode ser usado para receber entradas e interações onde elas são fornecidas pelo usuário.
* <Button>: Usado para criar um botão que os usuários podem tocar para realizar ações específicas, como especificado no exemplo: enviar um formulário.

A imagem abaixo mostra a estilização dos componentes anteriores, utilizando o StyleSheet, uma ferramenta que permite estilizar componentes, definindo estilos de forma eficiente e reutilizável, semelhante ao CSS em páginas da web.:

Figura 13 – Continuação formulário básico, Estilização

Tela preta com letras brancas

Descrição gerada automaticamente

Fonte: Do próprio autor, 2024

* const styles = StyleSheet.create({ container: { flex: 1, justifyContent: 'center', alignItems: 'center', padding: 20, backgroundColor: '#f0f0f0', }: Esse bloco, apresenta um estilo chamado container sendo definido. Ele configura o componente para ocupar todo o espaço disponível utilizando o (flex), centralizando verticalmente (justifyContent: 'center') quanto horizontalmente (alignItems: 'center'). Também, adiciona um preenchimento e define a cor de fundo (padding), (backgroundColor).
* title: { fontSize: 24, fontWeight: 'bold', marginBottom: 20, color: '#333', }: O Estilo é definido com o tamanho da fonte vinte e quatro (fontSize), em negrito (fontWeight: 'bold'). Ele adiciona uma margem inferior e define a cor do texto (marginBottom), (color).
* input: { width: '100%', marginBottom: 10, borderWidth: 1, borderColor: '#ccc', borderRadius: 5, padding: 12, fontSize: 16, backgroundColor: '#fff', }: A largura é definidada (width), uma margem inferior (marginBotton) e uma borda (borderWidht), com cor (borderColor) e cantos arredondados (borderRadius). O preenchimento interno e a fonte são estilizadas (padding), (fontsize). O fundo do campo é definido como branco (backgroundColor).
* messageInput: { height: 100, textAlignVertical: 'top', }, }): Por fim, é aplicado a um campo de entrada de mensagem um estilo. Ele define a altura (height) e alinha o texto verticalmente ao topo do campo (textAlignVertical: 'top').

Dessa maneira, ao ser executado em um dispositivo *mobile* utilizando o Expo, é apresentado o seguinte resultado:

Figura 14 – Resultado do aplicativo de formulário

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Fonte: Do próprio autor, 2024

Figura 15 – Resultado do aplicativo de formulário após pressionamento do botão

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Fonte: Do próprio autor, 2024

2.8 IoT

Mencionado por Santos (2019), a 'Internet das Coisas (IoT)' abrange as comunicações e processamentos de diversos equipamentos, incorporando uma integração de diversas tecnologias.

Diante do exposto, essa interação, é realizada captando eventos do mundo real e trocando informações entre si, possibilitando um controle do ambiente, conforme descrito pelo autor Sinclair (2018).

Em suma, a tecnologia vem revolucionando a maneira como a sociedade interage com as redes de dispositivos conectados, destacado por Magrani (2021), moldando as relações humanas e transformando os padrões de comunicação.

2.9 ESP32

O ESP32 é um microcontrolador que, de acordo com Santos (2019), é considerado barato e eficiente, com suporte a redes WI-FI e Bluetooth, possibilitando sua utilização em variadas execuções.

Conforme evidencia Zelenovsky e Mendonça (2017), os microcontroladores são dispositivos de pequeno porte programáveis, responsáveis pelo gerenciamento e execução de funções e componentes integrados a ele.

Segundo Morais (2023), esse microcontrolador exibe um grande poder de processamento e memória, podendo realizar aplicações com maior complexidade, focando em alto desempenho e conectividade.

Figura 16 – Esp32

Circuito eletrônico com fios

Descrição gerada automaticamente com confiança média

Fonte: Do próprio autor, 2024

2.10 Sensor MAX30102

Responsável pela medição da frequência cardíaca, o MAX30102, como explica Pascoal (2020), é um oxímetro e sensor de batimentos que prioriza a utilização em projetos portáteis, minimizando seu tamanho e mantendo a eficiência.

Figura 17 – Sensor MAX30102

Uma imagem contendo celular, telefone, velho, relógio

Descrição gerada automaticamente

Fonte: Do próprio autor, 2024

2.11 Battery Shield V8

Mencionado por Santos (2021), a Battery Shield V8 é uma *shield* eletrônico que integra duas baterias recarregáveis. Essas baterias oferecem um fornecimento de 6800 mAh e 3,7 V, proporcionando uma fonte de energia confiável e duradoura para o esp32.

2.12 C

O C é uma linguagem de programação de alto nível criada por Dennis Ritchie, de uso geral, portável e multiplataforma, permitindo a criação de programas complexos e acessíveis a todos os tipos de máquinas, como explicado por Damas (2016).

De grande popularidade, mesmo considerada de difícil aprendizado, possui influência em diversas linguagens diferentes, como o Java, PHP e C++, aponta Backes (2023).

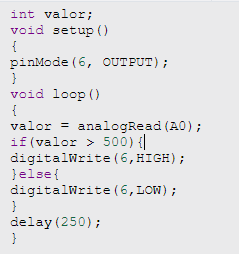
2.13 C++

Segundo Filho (2010), o C++ é uma linguagem que tem como base o C, considerada um subconjunto dele, possuindo diversas melhorias, como a orientação a objeto, e facilitando a criação de aplicações.

Por ser uma linguagem de alto nível, a execução de seus programas exige um tradutor, responsável por torná-lo de baixo nível, linguagem presente nas máquinas, assim explica Aguilar (2008).

A seguinte imagem exibe um exemplo de um código desenvolvido com o C++ em um Arduino Uno:

Figura 18 – Programa para manipulação de LEDs



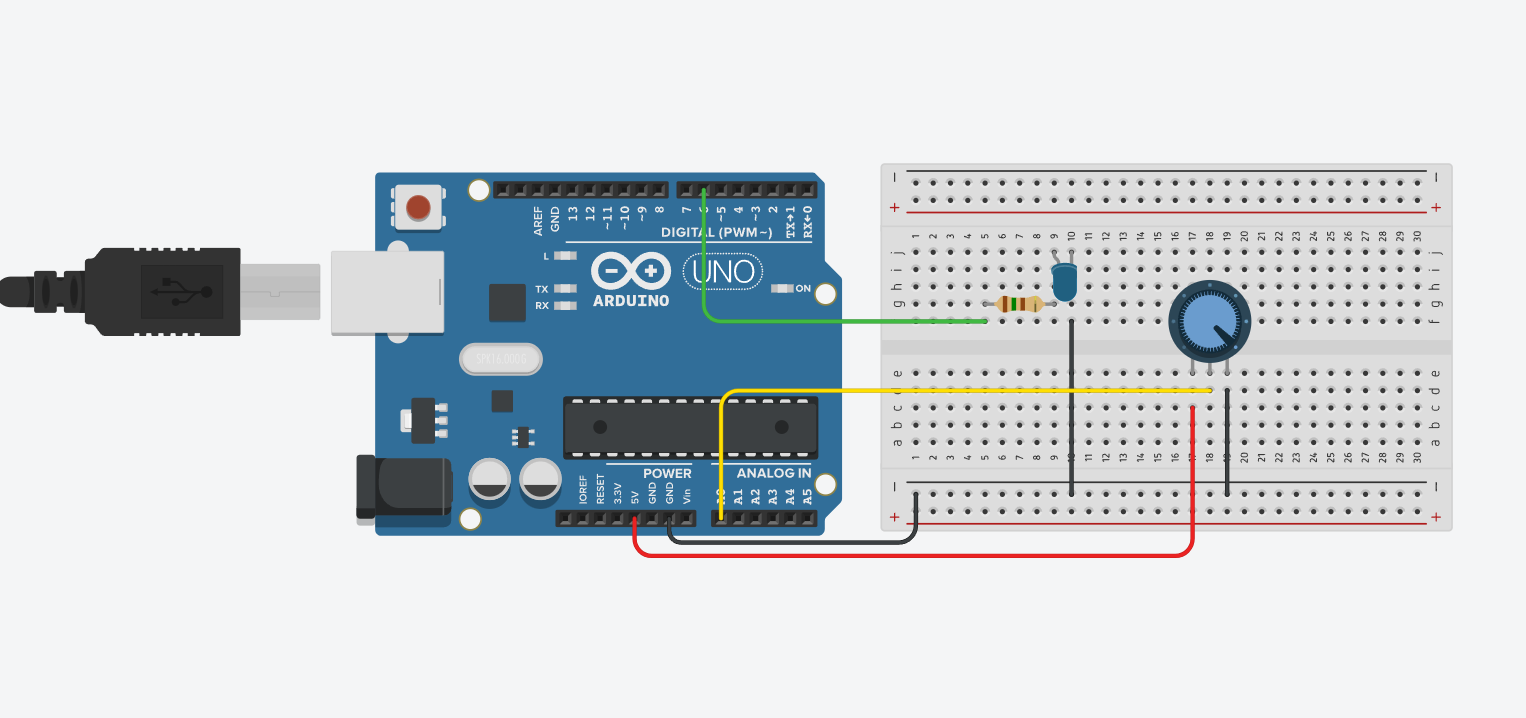
Fonte: do próprio autor, 2024

Esse exemplo é responsável pela manipulação de LEDs utilizando um potenciômetro, onde:

* int valor: declara uma variável, responsável por receber alguma informação.
* void setup(): função que declara configurações iniciais, como estado do LED, sendo ele ligado ou desligado.
* pinMode: declara qual porta será utilizada, além de definir como a informação será manipulada, seja ela uma entrada de dados ou saída.
* void loop(): função de ciclo responsável por executar todas as funcionalidades e comandos presentes nele, repetidamente.
* Valor = analogRead(A0): variável recebe a porta pertencente ao potenciômetro.
* *if*: bloco de decisão para limitar o valor do potenciômetro, onde é definido o estado do LED, se o valor for maior que 500, a lâmpada acende. Caso seja menor, ela irá desligar.
* digitalWrite(6, HIGH): define o LED como aceso.
* digitalWrite(6, LOW): torna o LED apagado.
* delay: tempo de espera para execução da ação.

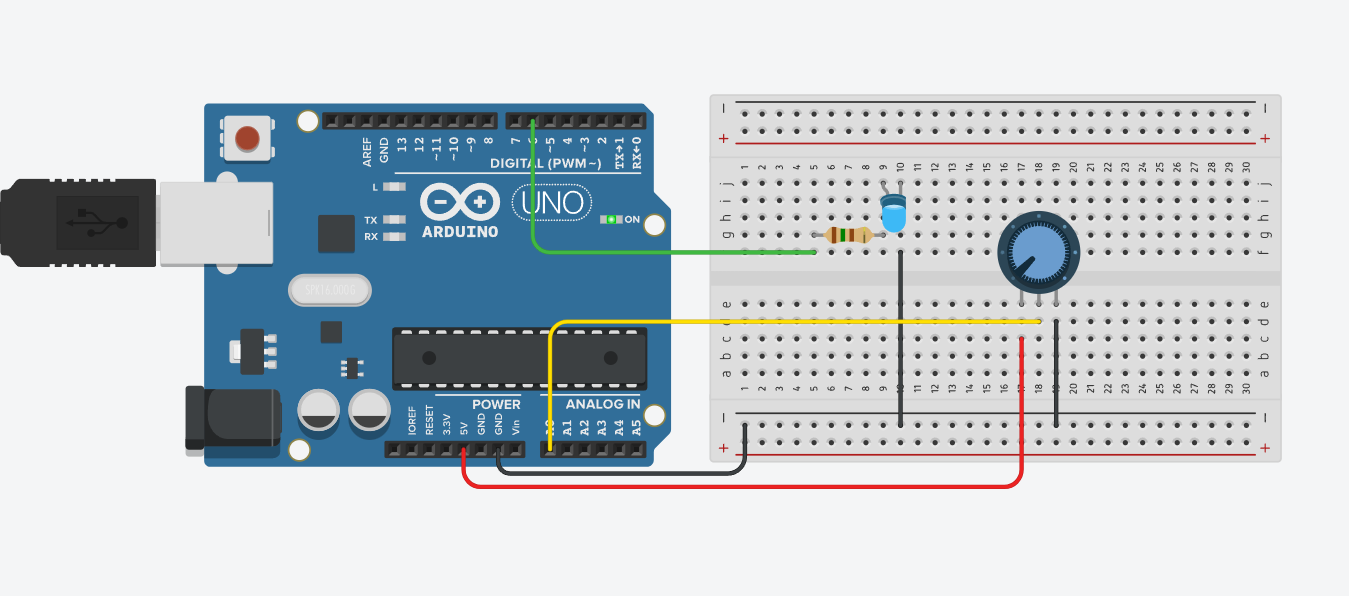
A figura abaixo exibe o resultado do código, onde o potenciômetro está desligado, deixando o LED apagado:

Figura 19 – Exemplo código C++ LED apagado

****Fonte: do próprio autor, 2024

A seguir, o potenciômetro encontra-se ligado, tornando o LED aceso:

Figura 20 – Exemplo código C++ LED aceso

Fonte: do próprio autor, 2024

2.17 Banco de dados

Um banco de dados, segundo Silberschatz, Korth e Sudarshan (2016), é considerado um grupo de informações organizadas com o objetivo de facilitar o acesso, controle e revisão desses dados.

Date (2004) insinua que as bases de dados são conhecidos por integrar os dados, diminuir as inconsistência e fornecer segurança aos dados. Os novos sistemas precisam dessas características para funcionar.

Elmasri e Navathe (2011) diz que os bancos de dados tanto relacionais, orientados a objetos e NoSQL foram desenvolvido para atender a diferentes requisitos e situações, oferecendo um leque de soluçoes caso necessario.

Nesse viez utilizamos a solução de um banco NoSQL.

Assim como lembrado por Sadalage e Fowler (2013), os bancos de dados NoSQL pertencem a uma subcategoria de sistemas de gerenciamento de banco de dados que não usam o modelo relacional convencional.

2.18 Firebase

Conforme mencionado pelo (Firebase, 2024), o serviço é uma plataforma de desenvolvimento de aplicativos fornecida pelo Google. Ela oferece uma variedade de serviços de nuvem e banco de dados, tanto para web quanto para dispositivos móveis.

O Firebase tem diversa bibliotecas específicas para cada ferramenta, simplificando o processo para os desenvolvedores, que só precisam identificar e instalar as bibliotecas necessárias como apresentado por Mezzari, Leal e Viegas (2019).

2.18.1 Firestore

Firestore é uma solução de banco de dados altamente versátil e dimensionável, parte integrante do Firebase. Utilizando uma abordagem não relacional ideal para o desenvolvimento de aplicativos móveis, como abordado por Araújo e Azevedo (2019).

Parafraseando Martins (2023), Ele desempenha um papel vital na gestão e armazenamento eficiente de informações. Graças a suas capacidades, é viável criar apps com escalabilidade e performance excepcionais.

Abaixo podemos observar uma ilustração da interface do Cloud Firestore:

Figura 21 – Exemplo de Estrutura do Firecloud

Tela de celular com aplicativo aberto

Descrição gerada automaticamente

Fonte: do próprio autor, 2024

Na imagem é possível observar três sessões, a primeira é o painel central, que mostra o banco atual selecionado e a opção de iniciar uma nova coleção, que podemos compreender como um grupo de documentos dentro de um banco de dados, sendo análoga a uma tabela em um banco de dados relacional. O segundo apresenta documentos dentro da coleção “usuários” selecionada anteriormente, documentos são uma unidade de armazenamento de dados com um código próprio, semelhante aos registros utilizados em unidades relacionais. Na terceira divisão é possível observar os campos exemplificado por “email”, “nome” e “senha”, seguido de seus respectivos valores, novamente semelhantes às colunas em uma tabela relacional. Observando essas funções é possível entender sua versatilidade e facilidade de uso para a criação de uma base de dados, facilitando os processos, anteriormente mais complexos em bancos de dados relacionas SQL.

Como relembrado por (Firebase, 2024), a ferramenta oferece suporte para regionalização do banco, disponibilizando um servidor em São Paulo, dessa maneira reduzindo a latência e aumentando a disponibilidade para projetos regionais.

Diante dessas informações optamos pelo Firebase, assim como o Cloud Firestore, devido a uma série de fatores. A plataforma é adequada para satisfazer as diversas e em constante mudança exigências do projeto, como apresentado por Oliveira (2023).

2.19 Impressão 3D

A impressão 3D, ou manufatura aditiva, é o método de criação de um objeto tridimensional utilizando materiais para sua produção, através de uma máquina responsável por processar essa matéria e moldar o produto, explica Volpato (2017).

2.20 Modelagem 3D

As impressões criadas são desenhadas previamente em um *software* específico para o desenvolvimento de peças tridimensionais. Esse processo de planejamento é chamado de modelagem 3D, assim conclui a Secretaria da Educação (2018).

Essas ferramentas, conhecidas como sistemas CAD (Computer Aided-Design), fornecem diversas utilidades para a facilitação da criação e visualização de todo o desenvolvimento dos componentes, como relata AutoDesk (2024).

2.21 Prototipação

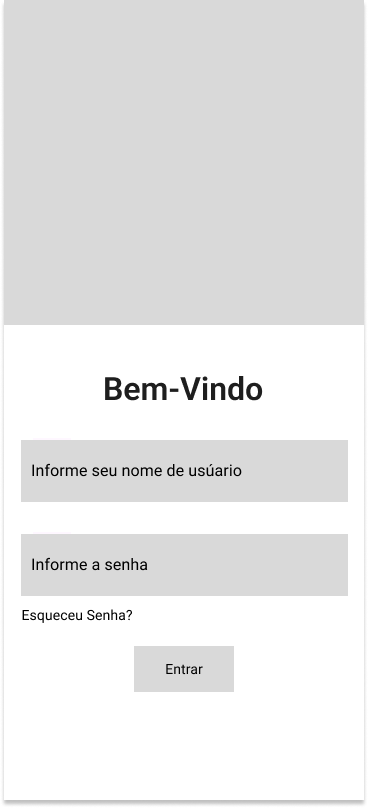
O autor Texeira (2014), informa que os *wireframes* são guias visuais utilizados para representar as estruturas das visualizações, assim como os elementos que as integram.

Descrito por [Busarello,](https://www.google.com.br/search?sca_esv=2012520a95282aaf&sca_upv=1&hl=pt-BR&sxsrf=ADLYWILsKOUeRHwBNJPqFpuQ_1ce_LH9Ow:1715821167913&q=inauthor:%22Raul+In%C3%A1cio+Busarello%22&tbm=bks) Bieging e Ulbricht (2013), tornam-se junções de ideias para a construção de protótipos das funcionalidades de tais componentes para a usabilidade do usuário.

Conforme destacado por Corrêa (2011), a primeira etapa da prototipação da interface envolve um visual simples (baixa fidelidade), seguido pelo refinamento do resultado com mais detalhes, podendo ser sua versão final (alta fidelidade).

Logo abaixo temos um exemplo de um Wireframe de Baixa Fidelidade:

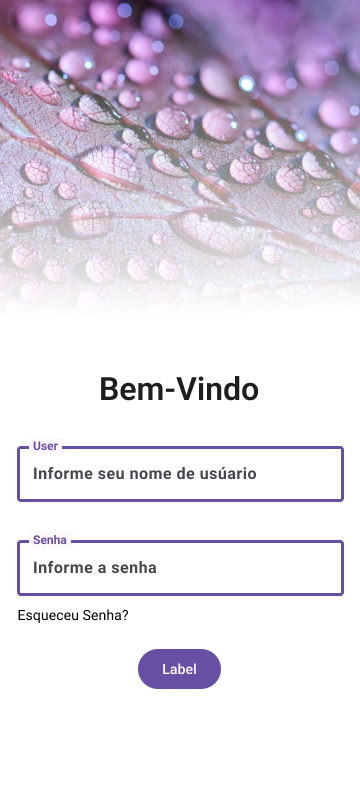
Figura 22 – Exemplo de Prototipação da Interface de Baixa Fidelidade



Fonte: do próprio autor, 2024

E em seguida, o Prototipação de Alta Fidelidade, ilustrando a versão final da aplicação:

Figura 23 – Exemplo de Prototipação da Interface de Alta Fidelidade



Fonte: do próprio autor, 2024

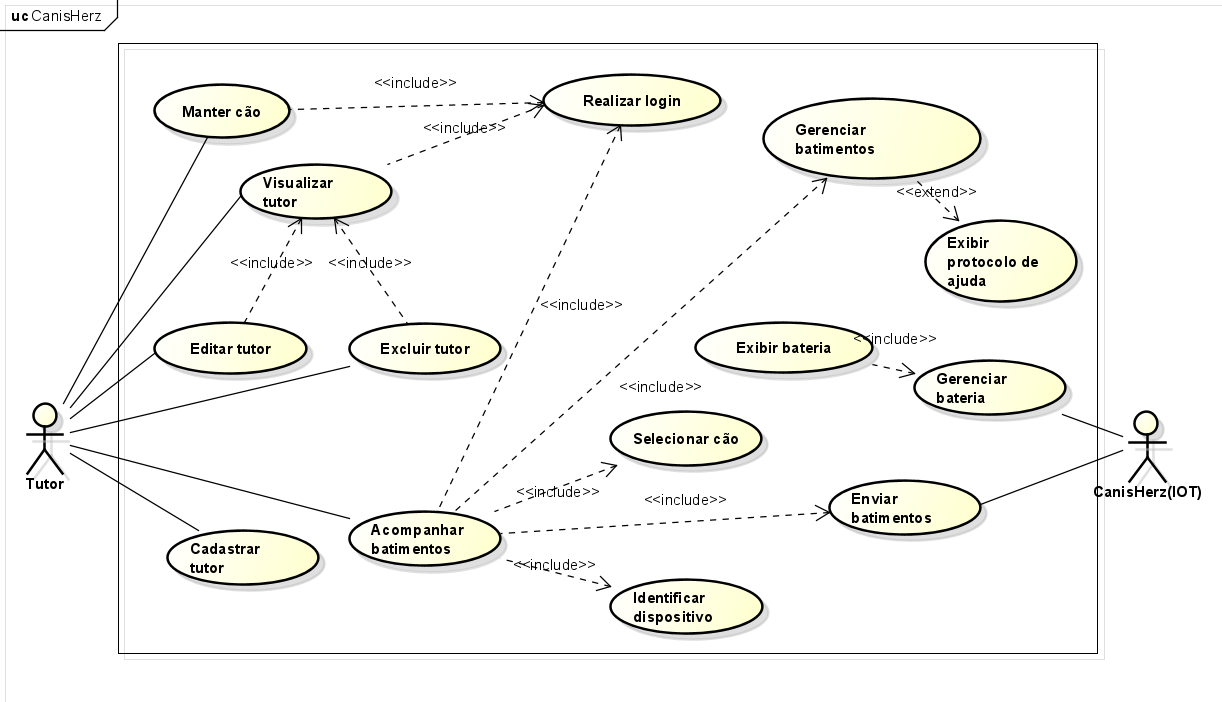
3 DESENVOLVIMENTO

O capítulo adiante apresenta o processo de prototipação e criação do sistema e dispositivo CanisHerz, utilizando métodos da UML. A seguir, serão introduzidos diagramas, protótipos de telas, imagens e processos físicos, demonstrando o passo a passo até a criação do produto.

3.1 Diagrama de Caso de Uso

A seguir, é possível observar uma imagem referente ao diagrama de caso de uso principal do aplicativo e dispositivo CanisHerz, apresentando as interações dos atores com o sistema e vice-versa, de maneira simples e direta, mostrando as ações que cada um deles, por sua vez, realizará no sistema.

Figura 24 – Diagrama de Caso de Uso CanisHerz



Fonte: do próprio autor, 2024

3.2 Documentação dos Casos de Uso

Seguindo o modelo UML, a seguir será introduzido a documentação dos casos de uso apresentados na imagem anterior, divididos por requisitos funcionais, não funcionais e regras de negócio.

Os requisitos funcionais representam as funcionalidades essenciais do sistema. Por outro lado, os requisitos não funcionais são características adaptáveis às preferências do cliente, relacionadas a aspectos como visual, desempenho e confiabilidade. Embora não sejam essenciais, são altamente desejáveis. As regras de negócio, por sua vez, são particularidades específicas aplicadas à venda do produto e aos serviços que serão oferecidos.

Requisitos Funcionais tutor:

* RF01: Cadastrar Tutor
* RF02: Realizar login
* RF03: Visualizar Tutor
* RF04: Editar Tutor
* RF05: Excluir Tutor
* RF06: Manter Cão
* RF07: Identificar Dispositivo
* RF08: Selecionar Cão
* RF09: Acompanhar Batimentos

Requisitos Funcionais Sistema geral:

* RF01: Gerenciar Batimentos
* RF02: Exibir protocolo de ajuda
* RF02: Exibir bateria

Requisitos Funcionais Iot(CanisHerz):

* RF03: Gerenciar Bateria
* RF04: Enviar batimentos

Requisitos Não funcionais Sistema Geral:

* RNF1: O dispositivo deve ser leve e confortável para o cão, evitando causar desconforto ou restrições de movimento.
* RNF2: A medição da frequência cardíaca deve ser precisa com uma margem de erro mínima.
* RNF3: O sistema deve ter uma autonomia de bateria suficiente para pelo menos 24 horas de operação contínua.
* RNF4: O aplicativo deve ser intuitivo e responsivo, com tempos de resposta rápidos.
* RNF5: Os dados de saúde do cão devem ser armazenados de forma segura e protegidos contra acesso não autorizado

Regras de Negócio:

* RN1: Os dados de saúde do animal pertencem exclusivamente ao proprietário e não serão compartilhados sem autorização explícita.
* RN2: O sistema não substitui a consulta a um veterinário profissional; todas as orientações fornecidas pelo aplicativo são apenas para referência e conscientização.
* RN3: O dispositivo e o aplicativo devem estar em conformidade com regulamentos locais e nacionais de proteção de dados e saúde animal.
* RN4: O sistema não permite afiliações de conta, porém é possível cadastrar mais de um cachorro em um único dispositivo.
* RN5: Não nos responsabilizamos por danos específicos causados ao animal, incluindo o mau uso do produto, conduta inadequada em primeiros socorros e complicações futuras não relacionadas ao sistema.

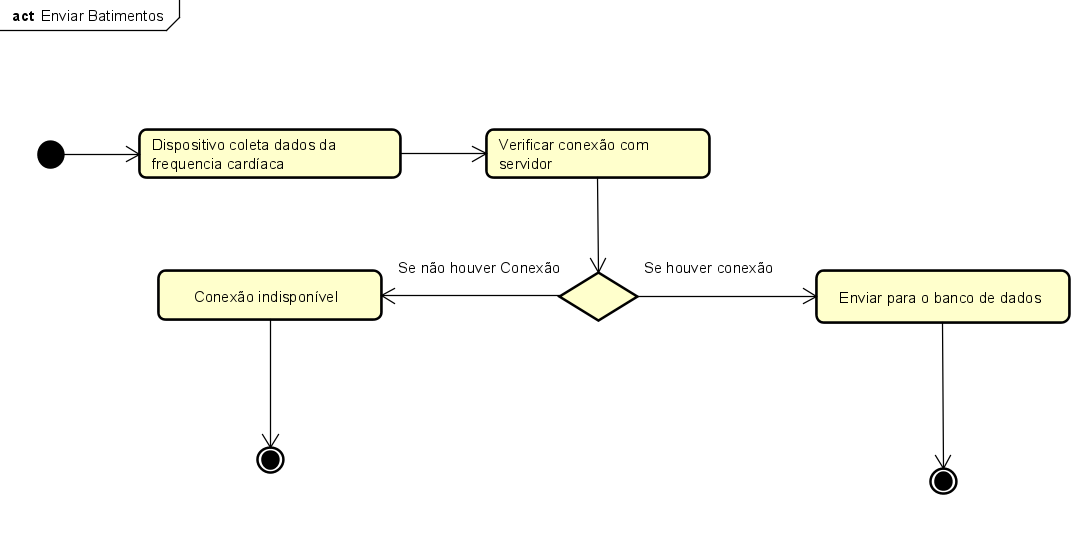
Descrição de caso de uso:

Falta Fazer

3.3 Diagrama de atividade

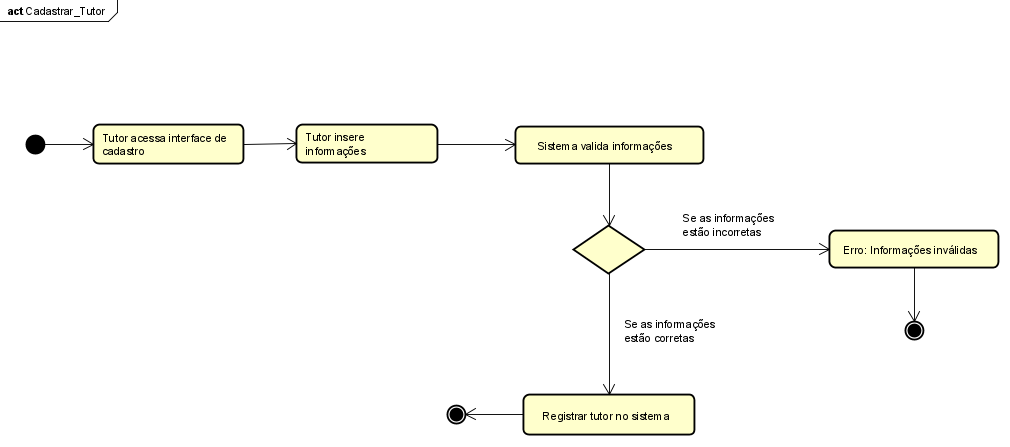
Os diagramas de atividade descrevem de forma abrangente a lógica e o fluxo de trabalho do sistema, mostrando o que cada processo implica sequencialmente. Cada diagrama fornece grandes detalhes sobre o fluxo de ações relacionadas a cada funcionalidade que pode ser acessada no sistema.

Figura 25 – Diagrama de Atividade: Enviar Batimentos



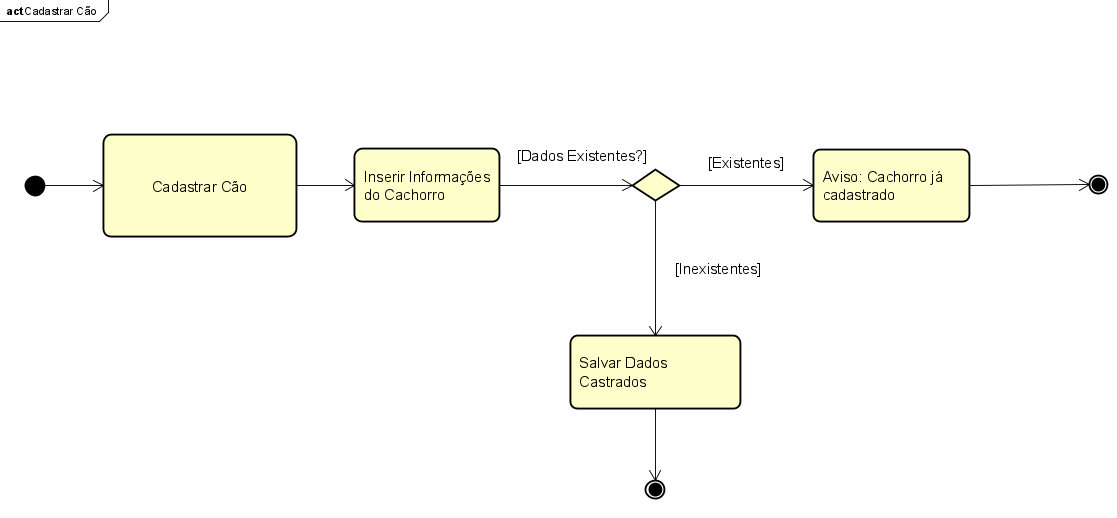
Fonte: do próprio autor, 2024

Figura 26 – Diagrama de Atividade: Cadastrar Tutor



Fonte: do próprio autor, 2024

Figura 27 – Diagrama de Atividade: Cadastrar Cão



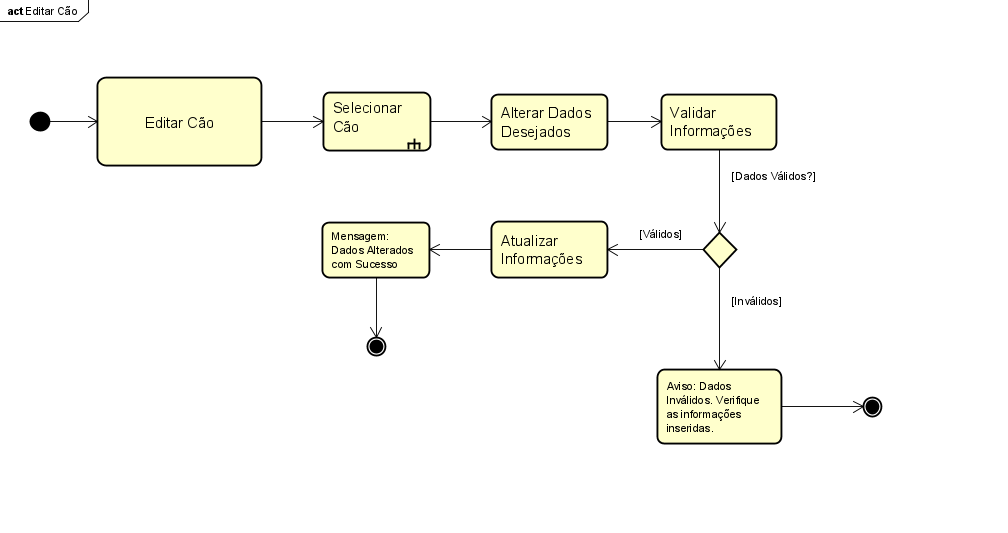
Fonte: do próprio autor, 2024

Figura 28 – Diagrama de Atividade: Selecionar Cão



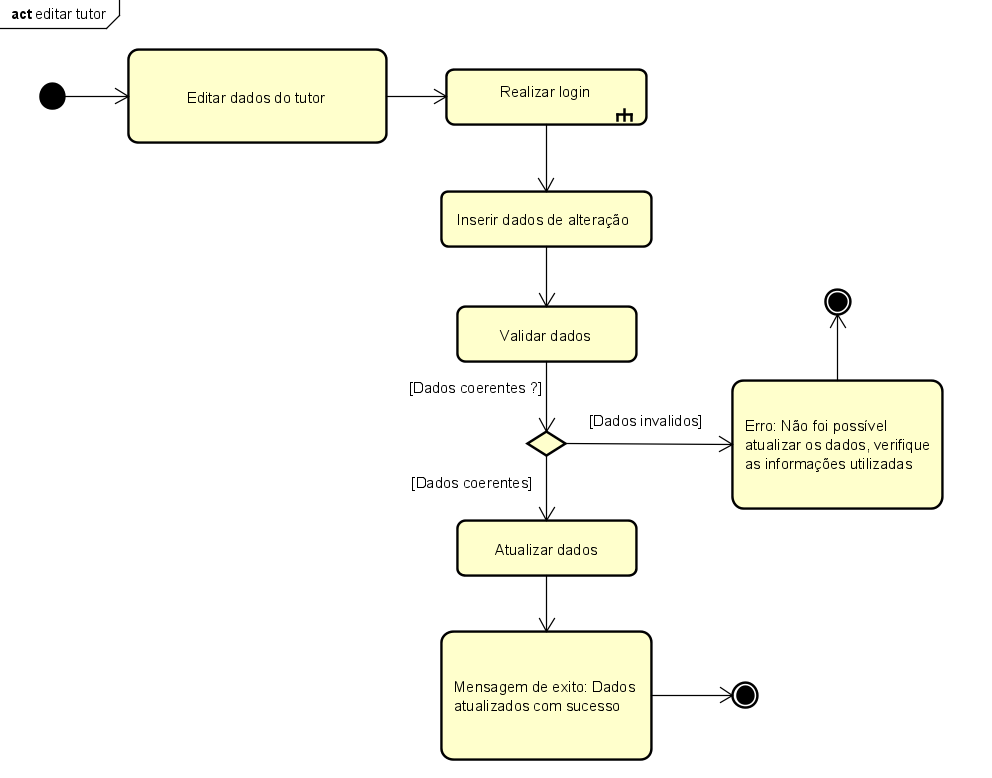
Fonte: do próprio autor, 2024

Figura 29 – Diagrama de Atividade: Editar Cão



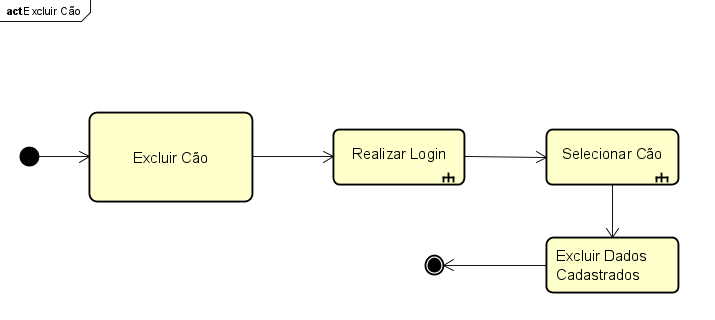
Fonte: do próprio autor, 2024

Figura 30 – Diagrama de Atividade: Editar Tutor



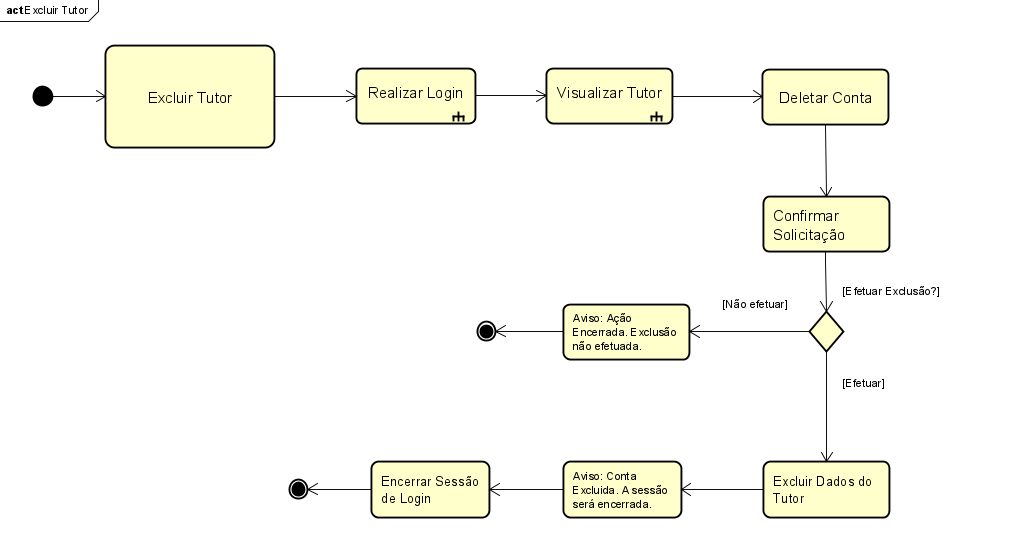
Fonte: do próprio autor, 2024

Figura 31 – Diagrama de Atividade: Excluir Cão



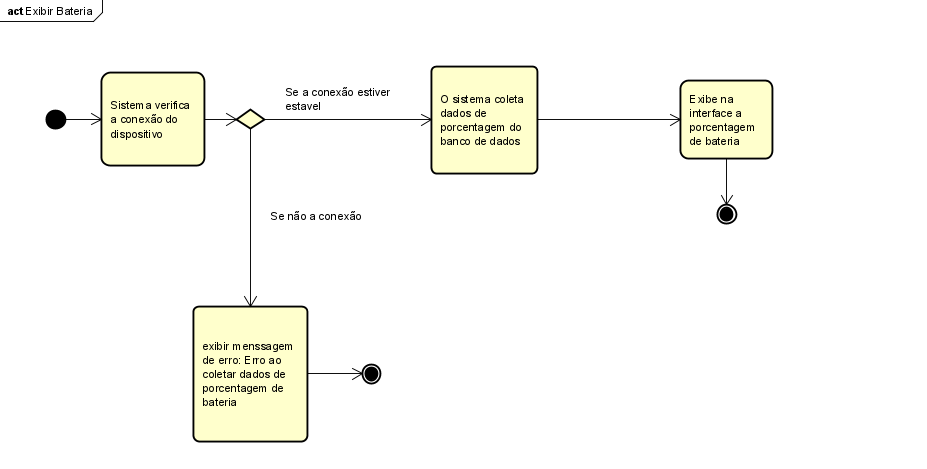
Fonte: do próprio autor, 2024

Figura 32 – Diagrama de Atividade: Excluir Tutor



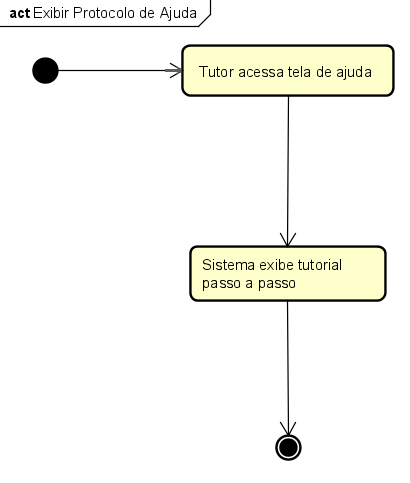
Fonte: do próprio autor, 2024

Figura 33 – Diagrama de Atividade: Exibir Bateria



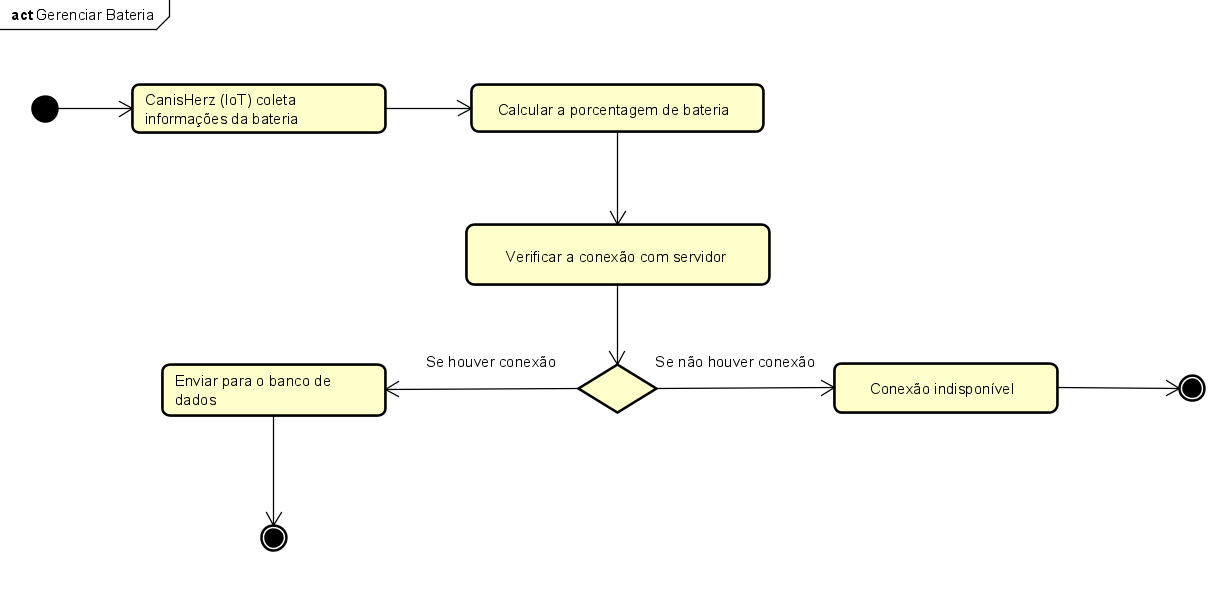
Fonte: do próprio autor, 2024

Figura 34 – Diagrama de Atividade: Exibir Protocolo de Ajuda



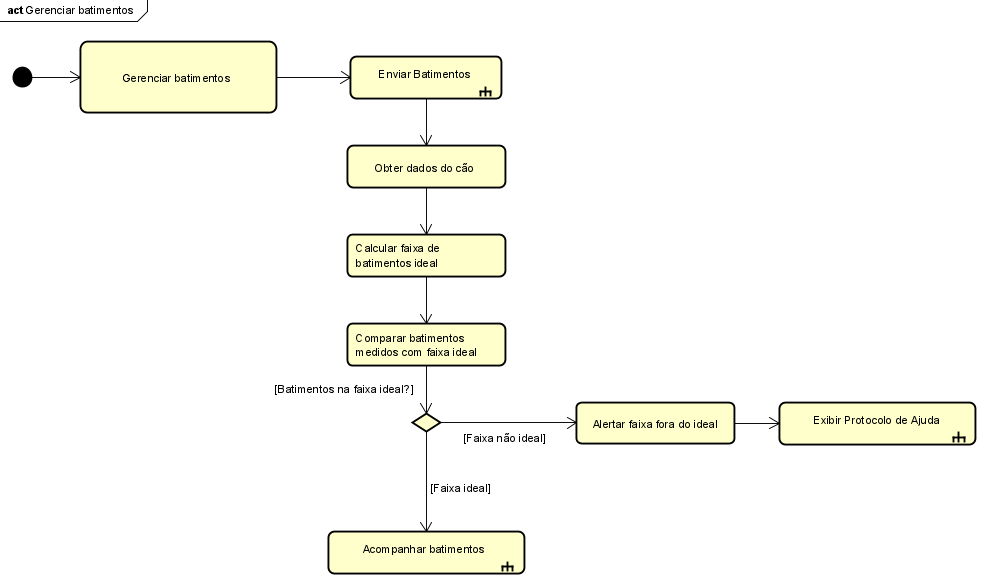
Fonte: do próprio autor, 2024

Figura 35 – Diagrama de Atividade: Gerenciar Bateria



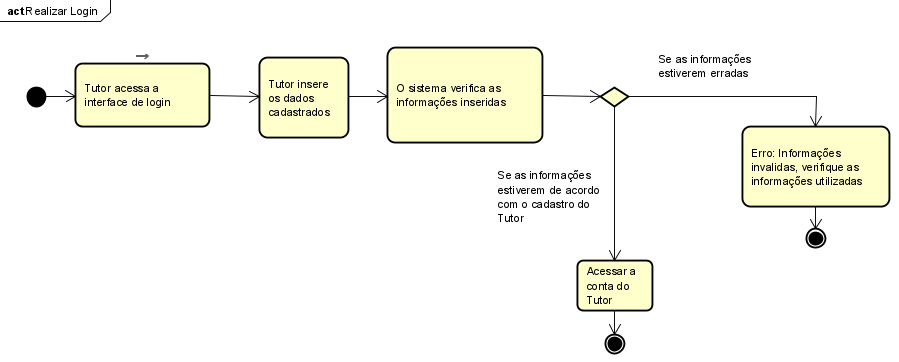
Fonte: do próprio autor, 2024

Figura 36 – Diagrama de Atividade: Gerenciar Batimentos



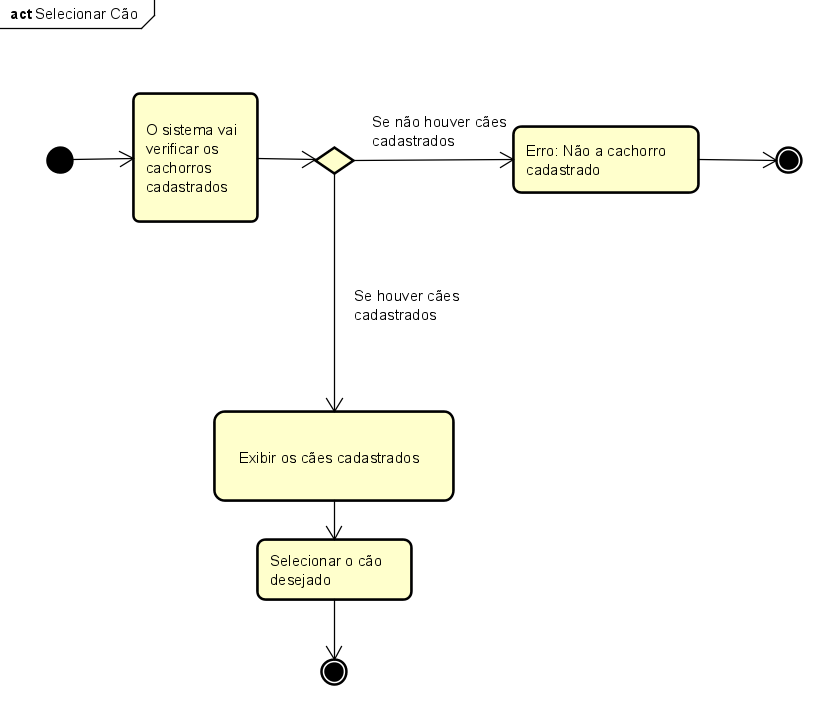
Fonte: do próprio autor, 2024

Figura 37 – Diagrama de Atividade: Realizar Login



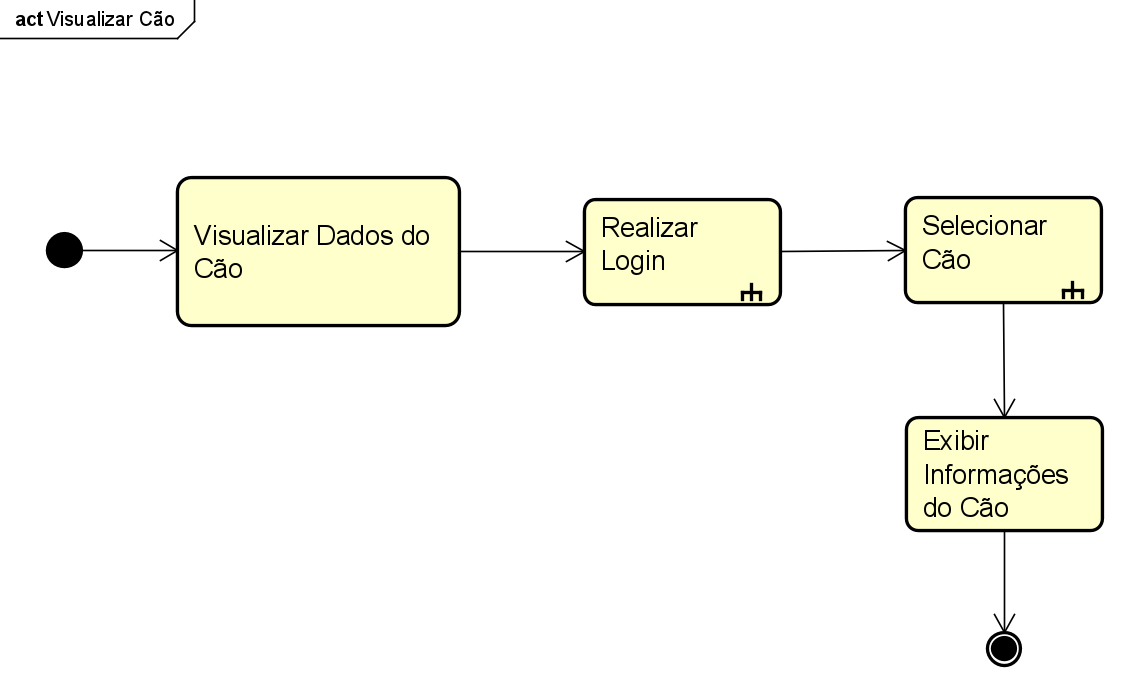
Fonte: do próprio autor, 2024

Figura 38 – Diagrama de Atividade: Selecionar Cão



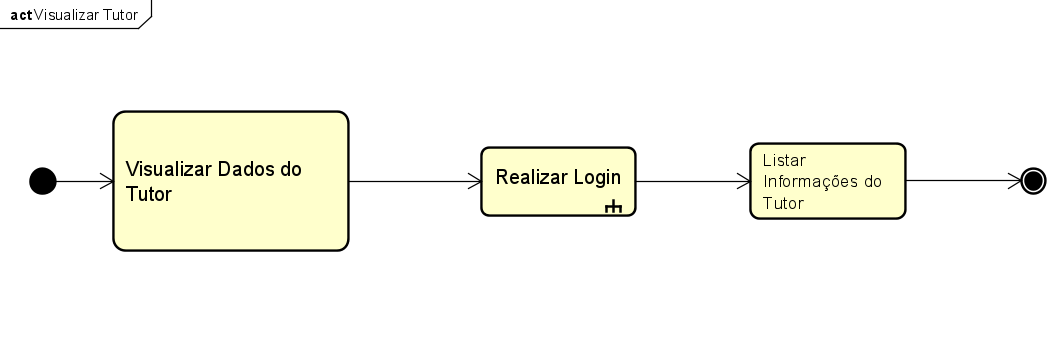
Fonte: do próprio autor, 2024

Figura 39 – Diagrama de Atividade: Visualizar Cão



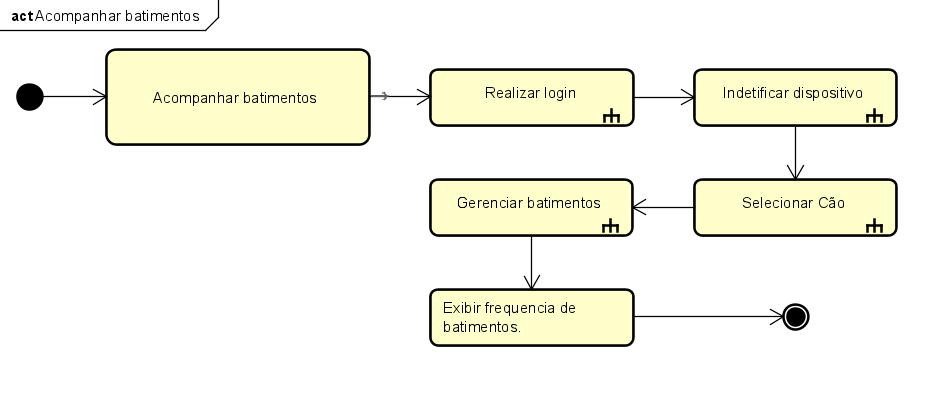
Fonte: do próprio autor, 2024

Figura 40 – Diagrama de Atividade: Visualizar Tutor



Fonte: do próprio autor, 2024

Figura 41 – Diagrama de Atividade: Acompanhar Batimentos



Fonte: do próprio autor, 2024

3.4 Diagrama de Sequência

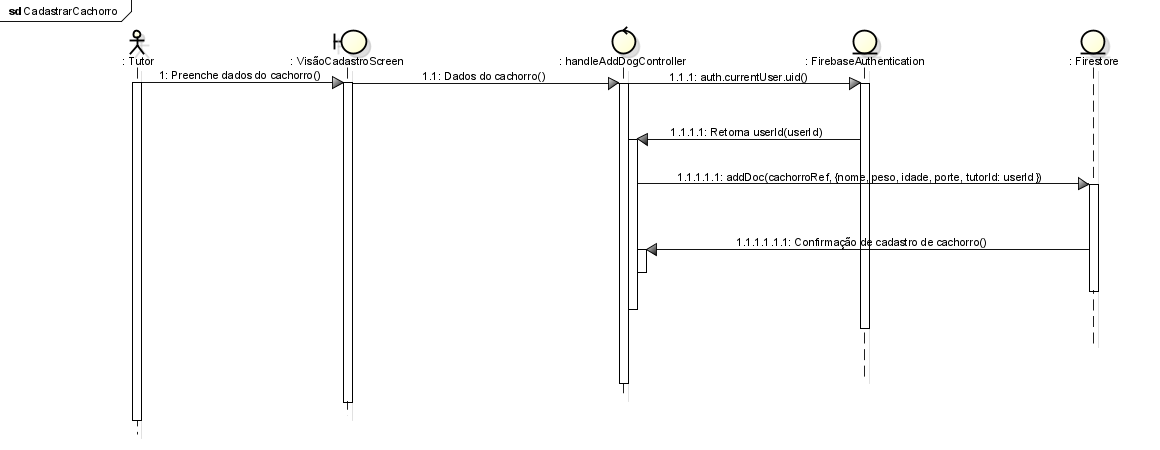
Com os Diagramas de Sequência, é possível compreender o comportamento de um único cenário, seguindo uma ordem temporal e as respectivas interações na execução de uma determinada função. Eles indicam a função que cada participante desempenha no sistema e a ordem em que cada interação deve ocorrer para a realização de uma ação específica no sistema.

Figura 42 – Diagrama de Sequência: Acompanhar Batimentos



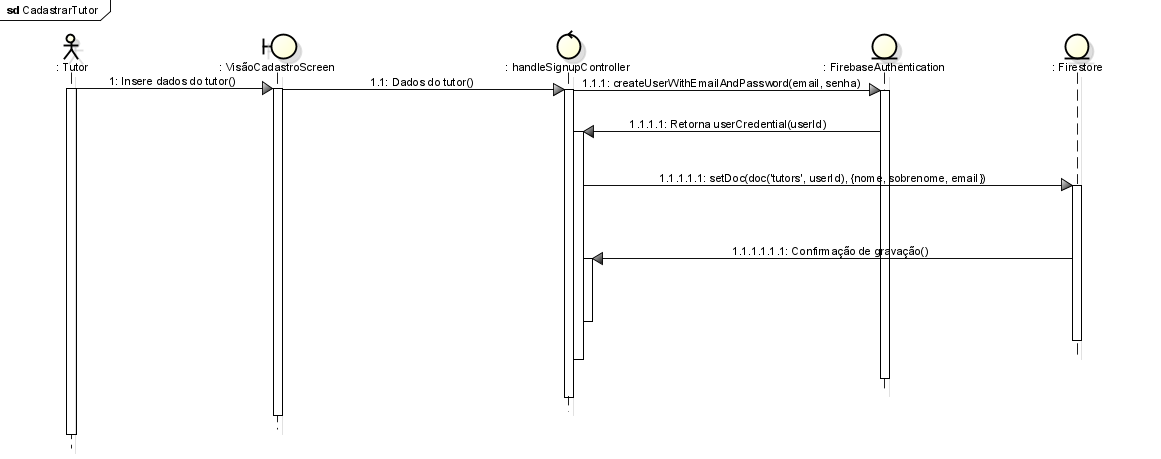
Fonte: do próprio autor, 2024

Figura 43 – Diagrama de Sequência: Cadastrar Cachorro



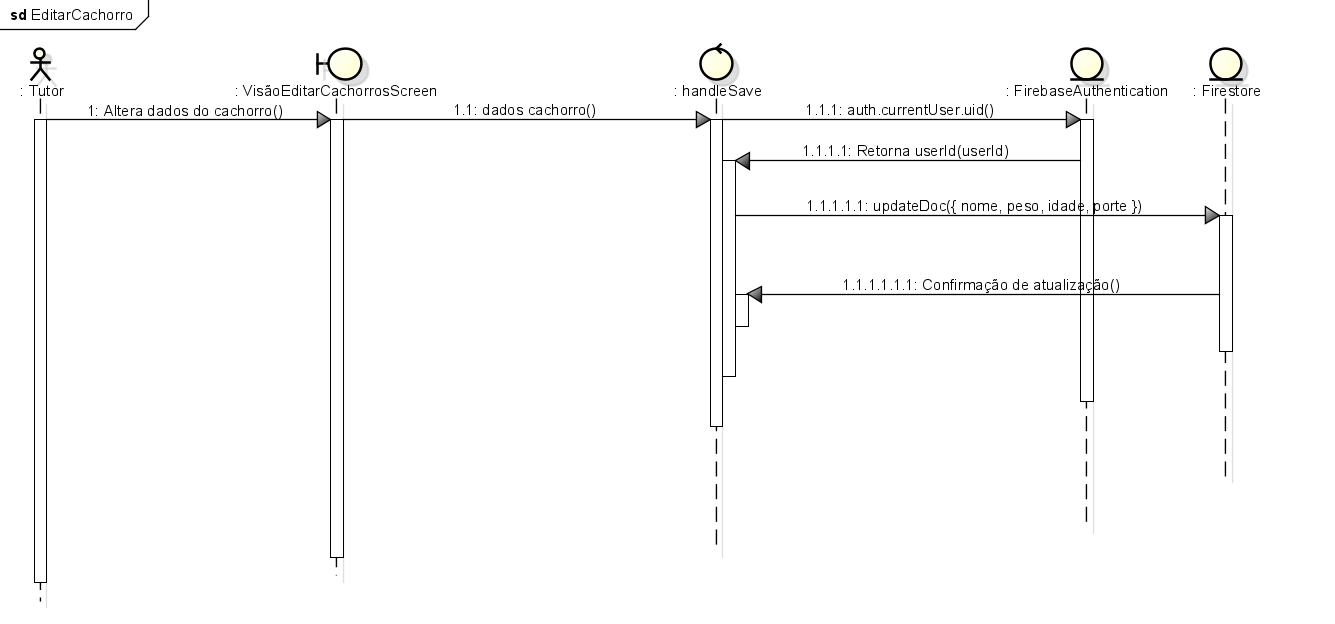
Fonte: do próprio autor, 2024

Figura 44 – Diagrama de Sequência: Cadastrar Tutor



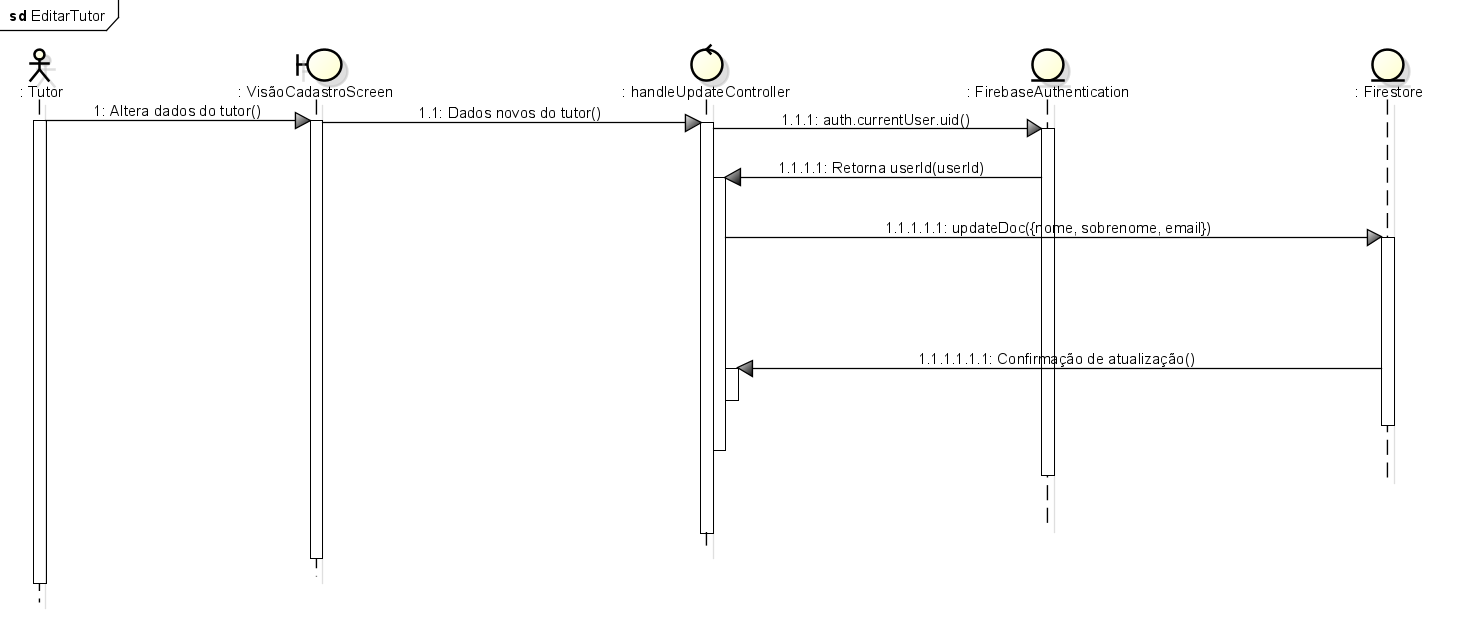
Fonte: do próprio autor, 2024

Figura 45 – Diagrama de Sequência: Editar Cachorro



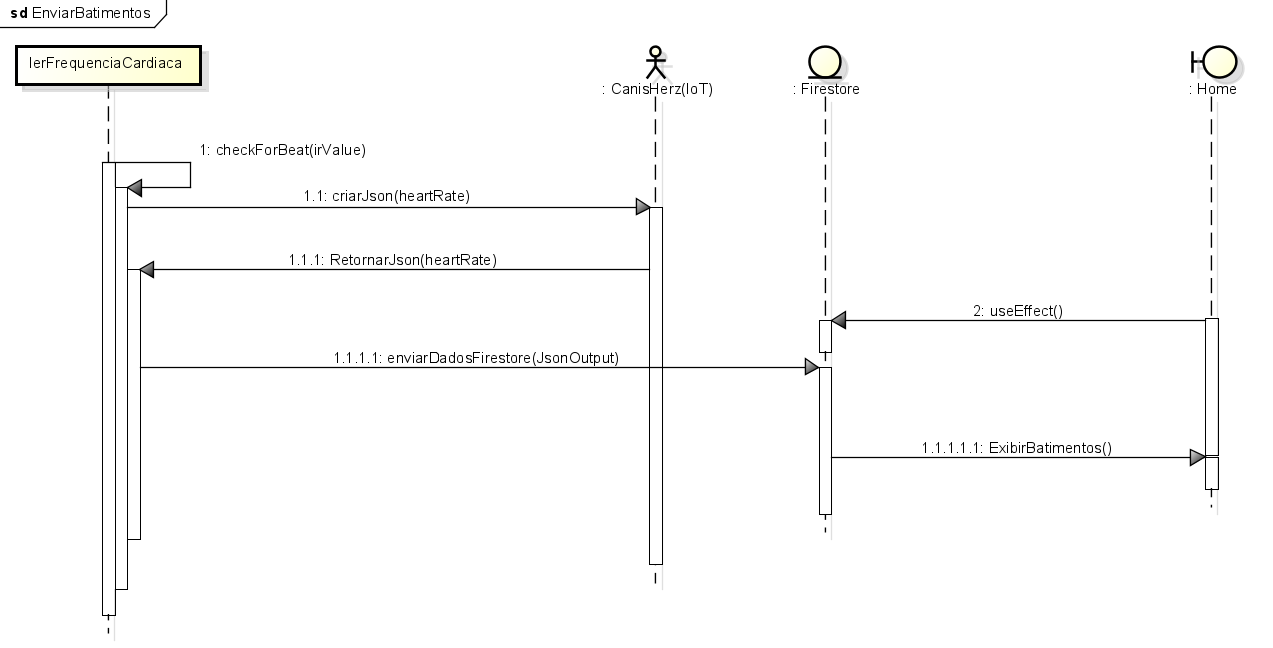
Fonte: do próprio autor, 2024

Figura 46 – Diagrama de Sequência: Editar Tutor



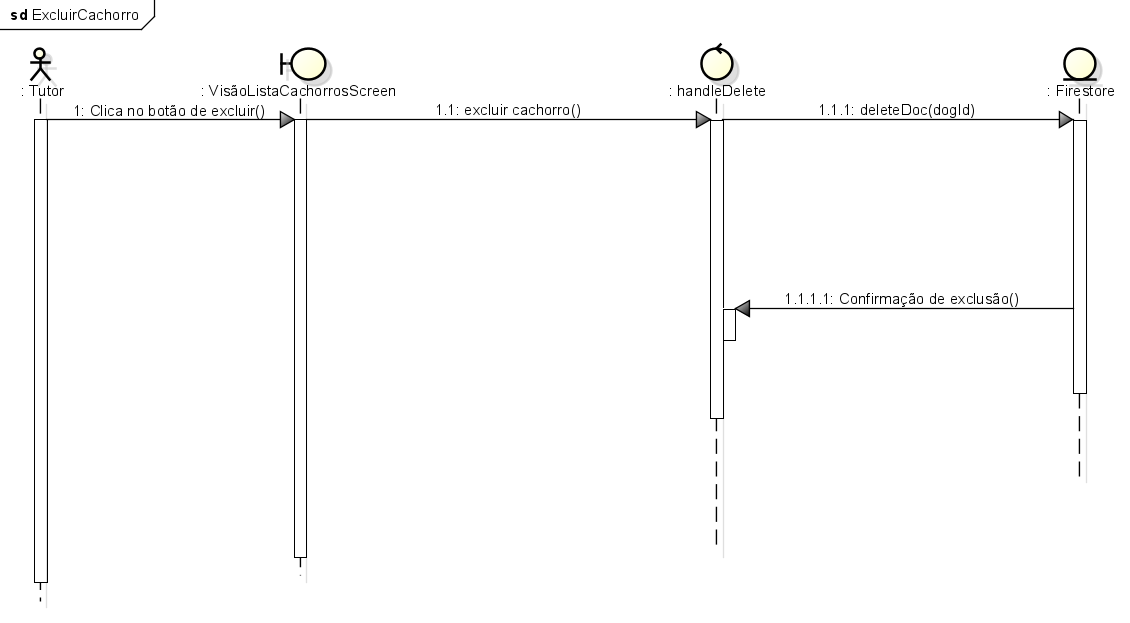
Fonte: do próprio autor, 2024

Figura 47 – Diagrama de Sequência: Enviar Batimentos



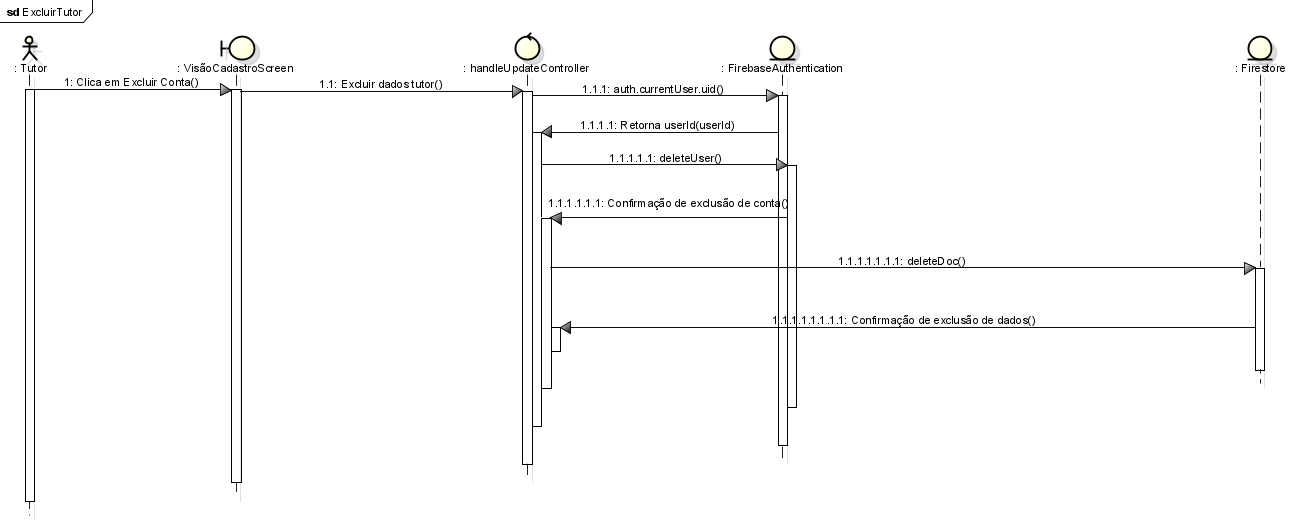
Fonte: do próprio autor, 2024

Figura 48 – Diagrama de Sequência: Excluir Cachorro



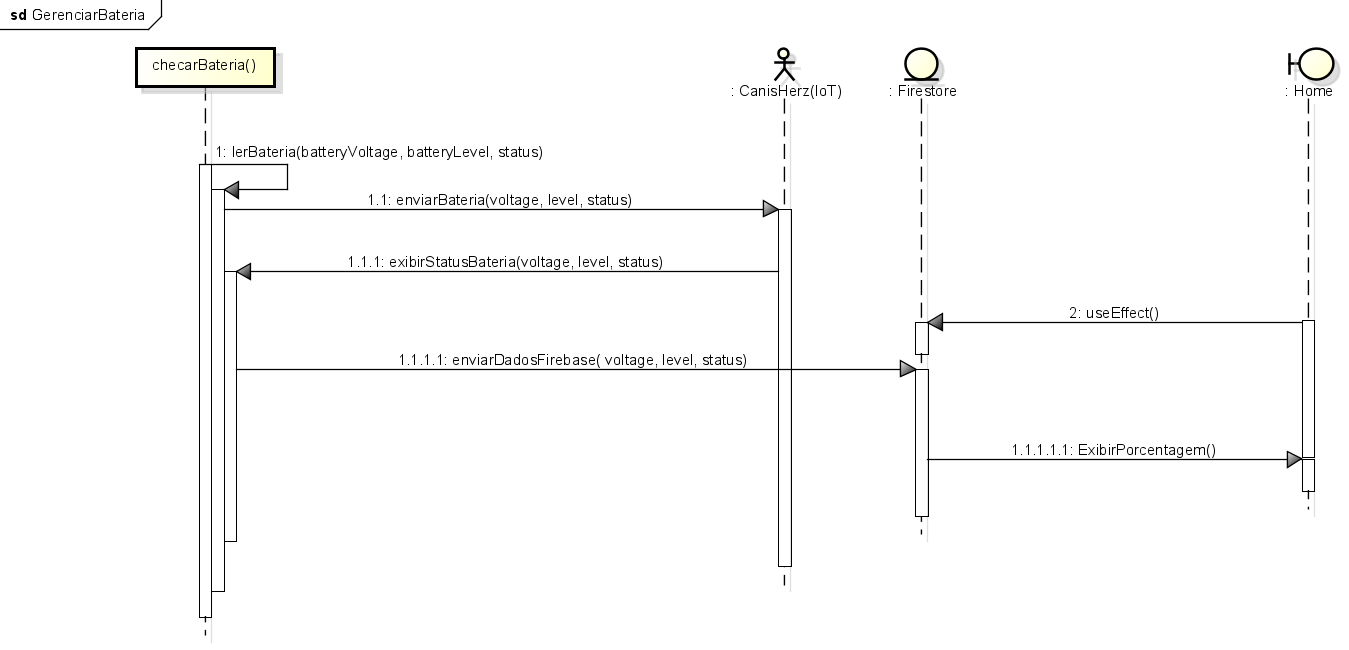
Fonte: do próprio autor, 2024

Figura 49 – Diagrama de Sequência: Excluir Tutor



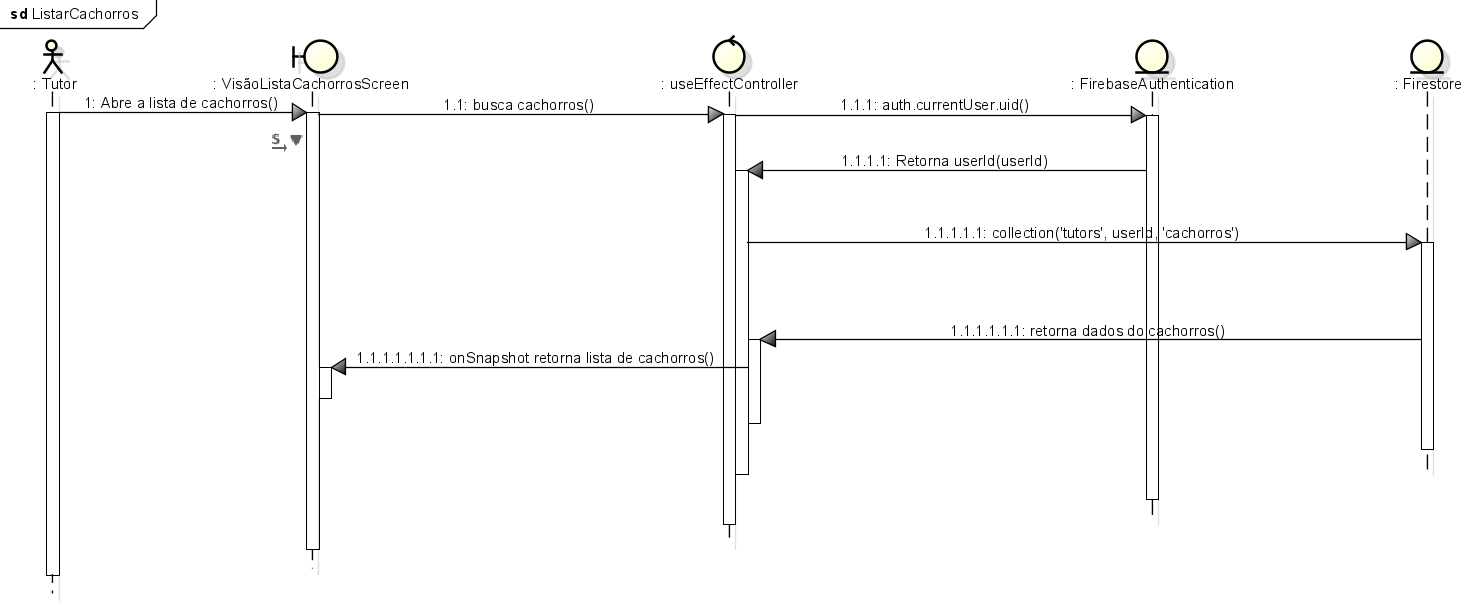
Fonte: do próprio autor, 2024

Figura 50 – Diagrama de Sequência: Excluir Tutor



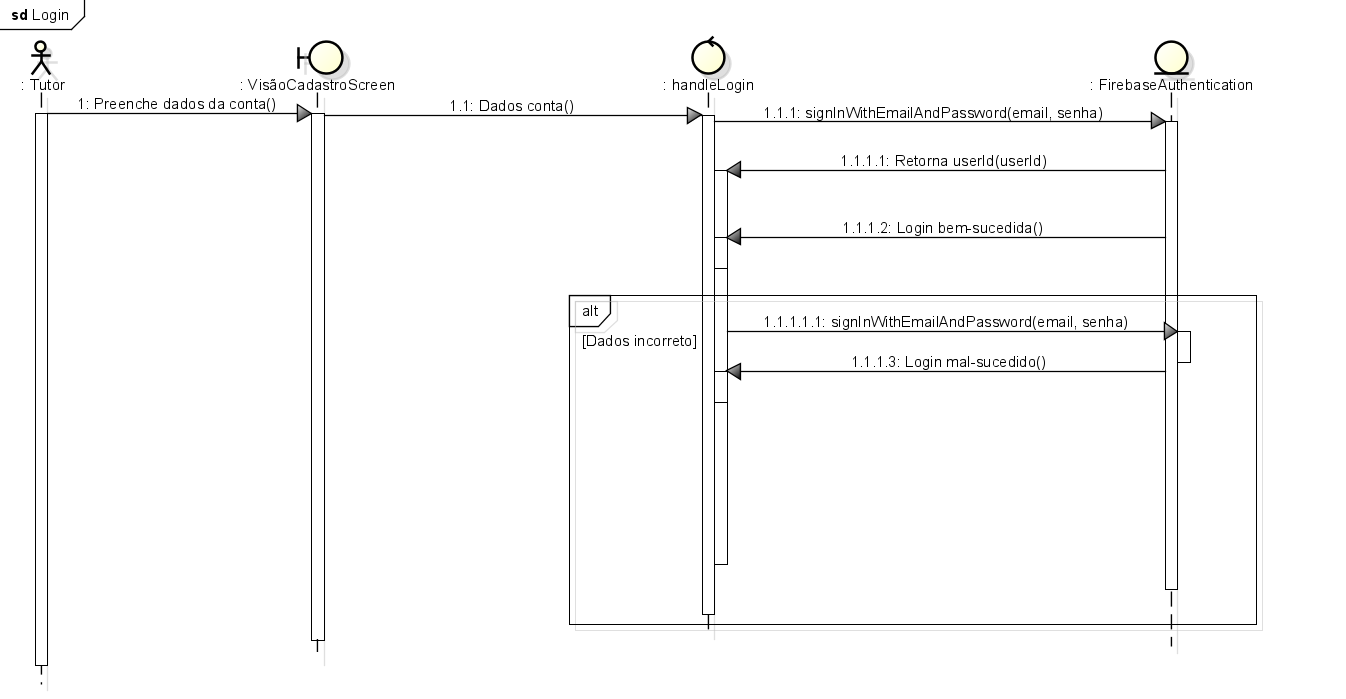
Fonte: do próprio autor, 2024

Figura 51 – Diagrama de Sequência: Listar Cachorros



Fonte: do próprio autor, 2024

Figura 52 – Diagrama de Sequência: Realizar Login



Fonte: do próprio autor, 2024

3.5 Diagrama de Máquina de Estado

O diagrama de máquina de estados, em geral é um diagrama usado na modelagem de sistemas orientados a objetos que descreve o comportamento dinâmico de uma entidade (objeto ou sistema) em resposta a eventos, sejam eles internos ou externos. Ele ilustra os estados pelos quais um sistema passa ao longo do tempo e como as transições entre esses estados ocorrem de acordo com eventos específicos.

Figura 53 – Diagrama de Máquina de estado: Autenticar Tutor

Diagrama

Descrição gerada automaticamente

Fonte: do próprio autor, 2024

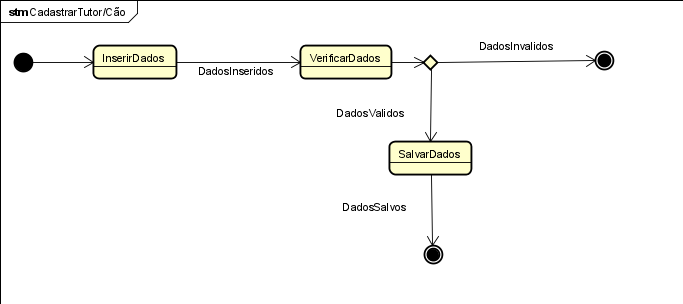
Figura 54 – Diagrama de Máquina de estado: Acompanhar Batimentos

Diagrama

Descrição gerada automaticamente

Fonte: do próprio autor, 2024

Figura 55 – Diagrama de Máquina de estado: Cadastrar Tutor/Cão



Fonte: do próprio autor, 2024

Figura 56 – Diagrama de Máquina de estado: Editar Tutor/Cão

Diagrama

Descrição gerada automaticamente

Fonte: do próprio autor, 2024

Figura 57 – Diagrama de Máquina de estado: Excluir Tutor/Cão

Diagrama

Descrição gerada automaticamente

Fonte: do próprio autor, 2024

Figura 58 – Diagrama de Máquina de estado: Exibir Bateria

Diagrama

Descrição gerada automaticamente

Fonte: do próprio autor, 2024

Figura 59 – Diagrama de Máquina de estado: Selecionar Cão

Interface gráfica do usuário, Diagrama, Texto

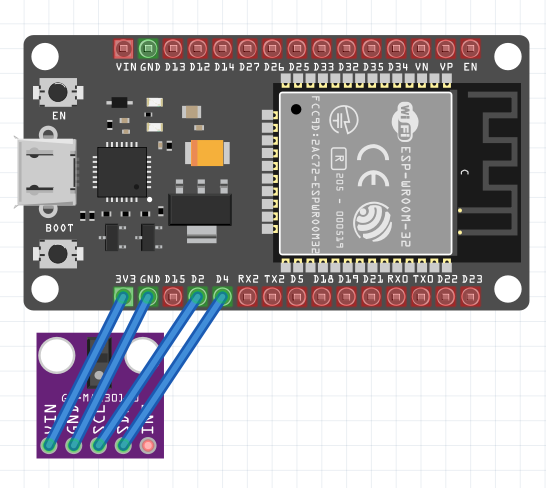
Descrição gerada automaticamente

Fonte: do próprio autor, 2024

3.6 Modelagem do sistema IoT

Esta seção fornecerá uma ilustração que descreve a inter-relação e a estrutura das partes elétricas no hardware físico do sistema Canis Herz. Ela inclui o componente ESP32 usado para o controle do aplicativo e o sensor MAX30102 usado para medir a circulação sanguínea. A ilustração mostrará como os cabos, as conexões e as vias de comunicação devem ser definidas entre os dispositivos, com ênfase especial na integração do microcontrolador ESP32 com o sensor MAX30102 na aquisição e transmissão das informações de frequência cardíaca do canino para o aplicativo.

Figura 60 – Esquemática de componentes Iot CanisHerz



Fonte: do próprio autor, 2024

O pino VCC VIN do MAX30102 é conectado ao pino de 3,3 V do ESP32:

* É um pino de alimentação de energia no sensor. O VIN/VCC é aqui fornecido pelo pino de 3,3 V do ESP32, que fornece a tensão necessária para operar o MAX30102.

O GND do MAX30102 é conectado ao GND do ESP32:

* o pino GND do sensor é conectado ao GND do ESP32 e, portanto, o circuito da fonte de alimentação está completo.

O pino SCL do MAX30102 é conectado ao D22 ou GPIO 22 do ESP32:

* Essa é a linha de relógio para fornecer o relógio para a comunicação I2C. Ela é conectada ao D22 GPIO 22 no ESP32 e funciona como um sinal de relógio para comunicação serial.

O SDA do MAX30102 é conectado ao pino D21 ou GPIO 21 do ESP32:

* O pino SDA é muito importante na comunicação I2C; esse pino envia e recebe dados pelo I2C, o que, em essência, faz com que o ESP32 envie e receba dados com o sensor corretamente.

3.7 Prototipação de telas da aplicação mobile

A seção a seguir abrange a prototipação de baixa e alta fidelidade para o aplicativo móvel. Esses esquemas mostram, por meio de imagens, como cada tela deve ser estruturada no aplicativo, descrevendo, portanto, os elementos que, por meio de sua construção visual, comporão e interagirão no design final da interface do usuário.

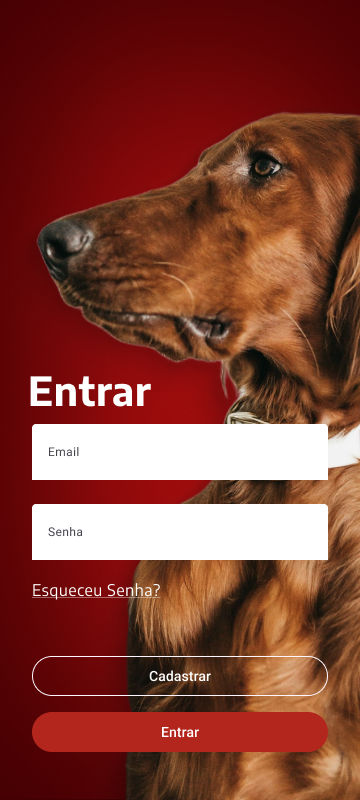
Abaixo as telas do aplicativo móvel, foram projetadas para abrigar uma interface de usuário desejável e tocante. A imagem de fundo é as fotos de cachorros, levam a ideia temática do aplicativo e aplicam um vínculo instantâneo com o usuário-alvo.

Figura 61 – Prototipação de tela: Boas-Vindas



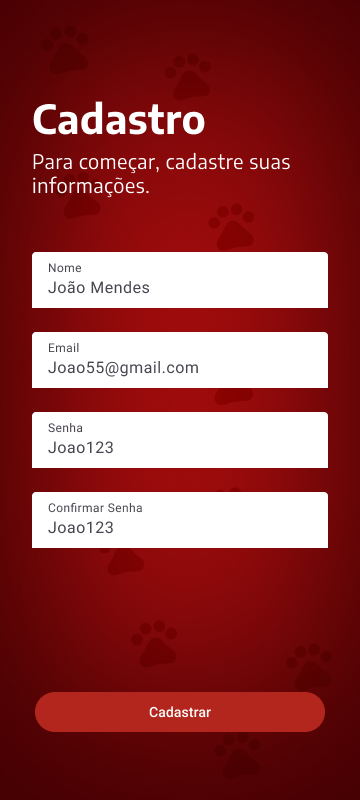
Fonte: do próprio autor, 2024

Figura 62 – Prototipação de tela: Entrar



Fonte: do próprio autor, 2024

Figura 63 – Prototipação de tela: Cadastrar Tutor



Fonte: do próprio autor, 2024

Figura 64 – Prototipação de tela: Cadastrar Cachorro



Fonte: do próprio autor, 2024

Figura 65 – Prototipação de tela: Proposito



Fonte: do próprio autor, 2024

Figura 66 – Prototipação de tela: Carregamento



Fonte: do próprio autor, 2024

Figura 67 – Prototipação de tela: Inicio (Sem conexão)



Fonte: do próprio autor, 2024

Figura 68 – Prototipação de tela: Inicio (Com conexão)



Fonte: do próprio autor, 2024

Figura 69 – Prototipação de tela: Medidor



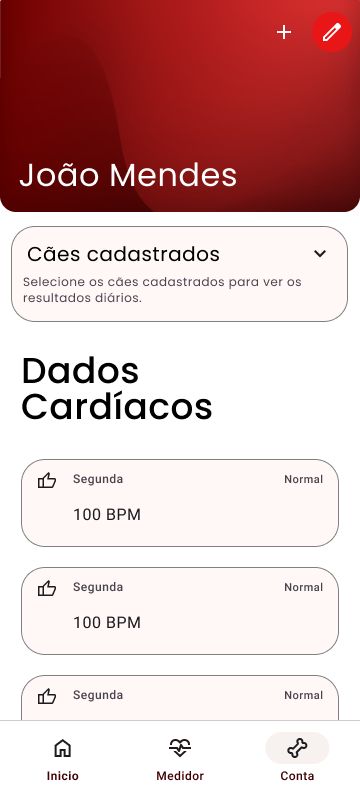
Fonte: do próprio autor, 2024

Figura 70 – Prototipação de tela: Medidor (Instruções fora da média)



Fonte: do próprio autor, 2024

Figura 71 – Prototipação de tela: Conta



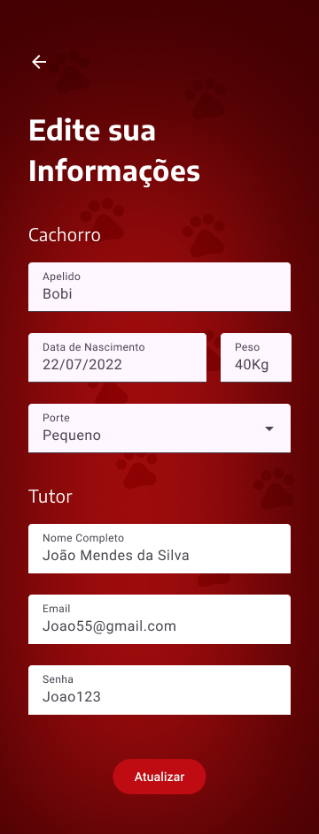
Fonte: do próprio autor, 2024

Figura 72 – Prototipação de tela: Novo Cachorro



Fonte: do próprio autor, 2024

Figura 73 – Prototipação de tela: Editar Informações



Fonte: do próprio autor, 2024

4 Conclusão

O esforço empreendido no projeto Canisherz firmou a convicção de que, sim, a tecnologia pode realmente contribuir muito para aprimorar a capacidade de monitorar a saúde dos animais de estimação, especialmente as doenças cardíacas caninas. O objetivo desse projeto era projetar um sistema com o qual seria possível monitorar os sinais vitais de um cão de forma não invasiva e em tempo real - uma ferramenta muito útil que ajudará veterinários, donos de animais e todos aqueles que se preocupam com o bem-estar de seus cães. O empreendimento comprovou o fato de que, quando a veterinária é combinada com a tecnologia, as condições cardíacas podem ser monitoradas sem muitos problemas, o que possibilita a detecção precoce de anormalidades de saúde. O sistema desenvolvido pela Canisherz funciona com base no princípio da captação constante dos batimentos cardíacos de um cão, que analisa a tendência e a armazena, de modo que o usuário possa passar por ciclos e identificar qualquer coisa incomum que exija um exame mais aprofundado. Isso não só melhoraria as verificações gerais de saúde, mas também poderia ser usado como uma forma precoce de prevenção, em que os problemas emergentes são capturados em um estágio inicial do ciclo. O sistema será, portanto, mais útil para lidar com condições crônicas e informar decisões sobre o cuidado e o bem-estar dos animais de estimação. Mais adiante, o objetivo é introduzir funcionalidades avançadas, como algoritmos de aprendizado de máquina, capazes de prever riscos futuros à saúde com base em dados anteriores. Dessa forma, o sistema estará em condições de emitir avisos detalhados que são muito mais eficazes como ferramenta de cuidados preventivos com a saúde. As versões futuras do dispositivo podem, portanto, ser concebidas para serem pequenas, flexíveis e fáceis de usar, de modo que os donos possam usá-lo em vários ambientes sem necessariamente ter de recorrer a um veterinário ou a um equipamento especial. Em outras palavras, a conceituação do Canisherz significa desenvolver o dispositivo para ser usado em qualquer lugar, inclusive ao ar livre, onde, na maioria das vezes, o pobre animal de estimação tem mais liberdade para se exercitar. Portanto, em princípio, toda a ideia do dispositivo trata da avaliação em tempo real de situações de variabilidade, que, faça chuva ou faça sol, não podem afetar seu trabalho. Desde que seja fácil de usar e portátil, o Canisherz funciona para tranquilizar os donos de animais de estimação, facilitando a capacidade de verificar a saúde de seus cães em qualquer lugar e a qualquer momento. Mas, mesmo como protótipo, o projeto Canisherz demonstrou um grande potencial ao provar que a tecnologia pode unir práticas veterinárias tradicionais e abordagens modernas baseadas em informações de forma bastante integrada. Foi estabelecida uma base sólida para a possível expansão do projeto, mesclando de forma eficaz a coleta de dados quantitativos com informações qualitativas de especialistas veterinários. Uma vez integrado a outros elementos de análise preditiva, maior facilidade de uso e compatibilidade com várias plataformas, o Canisherz poderá servir como uma ferramenta holística para o gerenciamento da saúde animal. A questão é que o Canisherz marca alguns quilômetros importantes na área de monitoramento da saúde de animais de estimação e pode, na verdade, mudar a forma como cuidamos de nossos cães. Embora ainda haja muito trabalho a ser feito, os resultados na fase inicial podem ser considerados bons. O aprimoramento contínuo dessa tecnologia pode permitir que um dispositivo Canisherz crie uma revolução nos cuidados com animais de estimação, em que a previsão precisa e em tempo real seja a base para o gerenciamento diário de cada proprietário de animal de estimação. Dessa forma, pretende-se estimular mais pesquisa e desenvolvimento no domínio para trazer abordagens ainda mais inovadoras que possam ser usadas para melhorar a qualidade de vida dos animais de estimação e de seus tutores.

REFERENCIAS

AGUILAR, Luis Joyanes**. Programação em C ++:** Algoritmos, estruturas de dados e objetos. 2. ed. Porto Alegre: AMGH, 2011.

BACKES, André. **Linguagem C:** completa e descomplicada. 2. ed. Rio de Janeiro: GEN, LTC, 2023.

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. **UML:** Guia do Usuário. 2. ed. Rio de Janeiro: Elsevier, 2012.

BRASIL. Secretaria de Estado da Educação do Paraná. Superintendência da Educação, Departamento de Políticas e Tecnologias Educacionais. **Impressão 3D: imaginar, planejar e materializar.** Paraná, 2018.

BUSARELLO, Raul Inácio; BIEGING, Patrícia; ULBRICHT, Vania Ribas. **Mídia e educação:** novos olhares para a aprendizagem sem fronteiras. São Paulo: Pimenta Cultural, 2013.

CARRIL, Marly. **HTML:** Passo a Passo. Joinville: Clube de Autores, 2012.

DAMAS, Luís. **Linguagem C.** 10. ed. Rio de Janeiro: GEN, LTC, 2016.

DATE, Christopher John. **Introdução a Sistemas de Bancos de Dados.** 8. ed. Rio de Janeiro: Elsevier, 2004.

ELMASRI, Ramez; NAVATHE, Shamkant B. **Sistemas de Bancos de Dados.** 6. ed. São Paulo: Pearson Addison Wesley, 2011.

ESCUDELARIO, Bruna de Freitas; PINHO, Diego Martins. **React Native:** Desenvolvimento de Aplicativos mobile com React. 2020. E-book.

FILHO, Antonio Mendes da Silva. **Introdução à programação orientada a objetos com C++.** Rio de Janeiro: Elsevier, 2010.

FLANAGAN, David. **JavaScript: o guia definitivo**. 6. ed. Porto Alegre: Bookman, 2013.

FLATSCHART, Fábio. **HTML 5:** Embarque Imediato. Rio de Janeiro: Brasport, 2011.

FOWLER, Martin et al. **UML Essencial:** Um breve guia para a linguagem-padrão de modelagem de objetos. 3. ed. Porto Alegre: Bookman, 2005.

GUEDES, Guilleanes Thorwald Araujo. **UML 2:** Uma Abordagem Prática. 3. ed. São Paulo: Novatec Editora, 2018.

JERICÓ, Márcia Marques; NETO, João Pedro de Andrade; KOGIKA, Márcia Mery. **Tratamento de Medicina Interna de Cães e Gatos**. Rio de Janeiro: Roca, 2015.

JOBSTRAIBIZER, Flávia. **Criação de Sites com CSS:** Desenvolva páginas Web mais leves e dinâmicas em menos tempo. São Paulo: Digerati Books, 2009.

JOBSTRAIBIZER, Flávia. **Criação de sites com o CSS:** Desenvolva páginas Web mais leves e dinâmicas em menos tempo**.** São Paulo: Digerati Books, 2009.

MAGRANI, Eduardo. **A Internet das Coisas.** Niterói: Cândido, 2021.

MILETTO, Evandro Manara; BERTAGNOLLI, Silvia de Castro. **Desenvolvimento de software II:** introdução ao desenvolvimento web com HTML, CSS, JavaScript e PHP. Porto Alegre: Bookman, 2014.

MORAIS, José V. M. **ESP32 com IDF**: O Guia Profissional. São Paulo: Instituto Newton C Braga, 2023.

MORRISON, Michael. **Use a cabeça! JavaScript.** Rio de Janeiro: Alta Books, 2008.

SADALAGE, Pramod J.; FOWLER, Martin. **NoSQL Essencial:** Um Guia Conciso para o Mundo Emergente da Persistência Poliglota. São Paulo: Novatec Editora Ltda, 2013.

SANTOS, Sandro. **Introdução à Iot:** Desvendando a Internet das Coisas. Joinville: Clube dos Editores, 2019.

SANTOS, Wallef Ferreira. E**quipamento de baixo custo para monitorar temperatura e umidade de forma contínua e remota:** aplicação na compostagem. 2023. 53 f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Ambiental e Sanitária) - Centro de Tecnologia, Universidade Federal de Alagoas, Maceió, 2021.

SCHEIDT, Felippe Alex. **Fundamentos de CSS:** Criando Design para Sistemas Web. Paraná: Outbox Interativa, 2015.

SILBERSCHATZ, Abraham; SUNDARSHAN, S.; KORTH, Henry F. Sistema de Banco de Dados. Rio de Janeiro: Elsevier, 2016.

SILVA, Maurício Samy. **Criando Sites com HTML:** Sites de Alta Qualidade com HTML e CSS. São Paulo: Novatec Editora Ltda, 2008.

SILVA, Maurício Samy. **React: Aprenda Praticando** - Desenvolva aplicações web reais com uso da biblioteca React e de seus módulos auxiliares. São Paulo: Novatec Editora Ltda, 2021.

SINCLAIR, Bruce. **IoT:** Como Usar a "Internet Das Coisas" Para Alavancar Seus Negócios. São Paulo: Autêntica Business, 2018.

TEIXEIRA, Fabrício. **Introdução e boas práticas em UX Design.** São Paulo: Casa do Código, 1998.

VOLPATO, Neri. **Manufatura Aditiva:** Tecnologias e aplicações da impressão 3D. São Paulo: Blucher, 2017.

ZAKAS, Nicholas Charles. **Princípios de Orientação a Objetos em JavaScript**. São Paulo: Novatec Editora, 2017.

ZELENOVSKY, Ricardo; MENDONÇA, Alexandre. **Microcontroladores:** Programação e Projeto com a família 8051. Rio de Janeiro: MZ Editora Ltda, 2017.

ARAÚJO, Guilherme Ianhes; AZEVEDO, Rodolfo Jardim. **Implementação de um Sistema de Quiz no Aplicativo WebLectures.** 2019. Relatório Técnico (Graduação em Computação) – Universidade Estadual de Campinas, Campinas, 2019.

BEZERRA, Franklyn Seabra Rogério; VIANA, Wilson. **Desenvolvimento Nativo vs Ionic vs React Native:** uma análise comparativa do suporte à acessibilidade em Android**.** 2021. TCC (Graduação em Computação) - Universidade Federal do Ceará, Fortaleza, 2021.

CORRÊA, Daniel Felippe Bernardino. **O Papel da Arquitetura de Informação na Experiência do Usuário.** 2011. TCC (Especialização em Arquitetura e Organização da Informação) – Universidade Federal de Minas Gerais, Belo Horizonte, 2011.

FRANCO, Matheus Felipe Souza. **Percepção de tutores sobre a saúde e qualidade de vida de cães cardiopatas.** 2022. TCC (Bacharelado em Medicina Veterinária) - Universidade Federal da Fronteira Sul, Realeza, 2022.

MARTINS, João Antônio Bandeira De Oliveira. **Uma ferramenta para otimizar a relação instrutor-aluno em academias.** 2023. TCC (Bacharelado em Ciência da Computação) – Universidade Federal de Campina Grande, Campina Grande, 2023.

OLIVEIRA, João Olívio Scaramussa Fávero. **Desenvolvimento de uma solução com gamificação para estudos de engenharia de requisitos.** 2023. TCC (Bacharelado em Sistemas da Informação) – Instituto Federal do Espírito Santo, Cachoeiro de Itapemirim, 2023.

PASCOAL, Pedro Gelati. **Desenvolvimento de um sistema para monitoramento da frequência cardíaca em atividades físicas.** 2020. TCC (Graduação em Engenharia Elétrica) – Universidade Regional do Noroeste do Estado do Rio Grande do Sul, Ijuí, 2020.

MEZZARI, Lucas Torres; LEAL, Eduardo Henrique Viva; VIEGAS, Silvio.Internet das Coisas: Arduino, Firebase e Android. **Gestão e Tecnologia**. Gravataí, v. 5, n. 1, set. 2019.

SILVA, Denys Alves; SOUSA, Caio Frias. Construção de App com React Native. **Tecnologias em Projeção**. Brasília, [v. 10, n. 1, ago.](https://revista.projecao.br/index.php/Projecao4/issue/view/106) 2019.

ALURA. **Como instalar e configurar o Expo do React Native.** Disponível em: <https://www.alura.com.br/artigos/como-instalar-configurar-expo-do-react-native>. Acesso em: 07 de mai. 2024

AUTODESK. **Software CAD para arquitetos, engenheiros e projetistas.** Disponível em: https://www.autodesk.com.br/solutions/cad-software. Acesso em: 16 de mai. 2024

CONSELHO REGIONAL DE MEDICINA VETERINÁRIA DO ESTADO DE SÃO PAULO. **Pets também podem ter problemas cardíacos.** CRMV-SP, 26 set. 2019. Disponível em: https://crmvsp.gov.br/pets-tambem-podem-ter-problemas-cardiacos/. Acesso em: 2 jun. 2024.

FELDMAN, Andre. **Cardiopatia**: o que é, sintomas, causas, tipos e tratamento. 2024. Disponível em: https://www.tuasaude.com/cardiopatia. Acesso em: 10 de abr. 2024

FIREBASE. **Armazene e sincronize dados em tempo real.** Disponível em: https://firebase.google.com/products/realtime-database?utm\_source=bing&utm\_medium=cpc&utm\_campaign=latam-BR-all-pt-dr-SKWS-all-all-trial-e-dr-1707800-LUAC0016441&utm\_content=text-ad-none-any-DEV\_c-CRE\_-ADGP\_Hybrid%20%7C%20SKWS%20-%20MIX%20%7C%20Txt\_%20Compute-Firebase-KWID\_43700067403163247-kwd-78684157082834%3Aloc-20&utm\_term=KW\_Firebase-ST\_Firebase&hl=pt-br. Acesso em: 27 de abr. 2024

FRAGATA, Fernanda. 10 sintomas para detectar doenças cardíacas em cães. Disponível em: https://epoca.globo.com/colunas-e-blogs/fernanda-fragata/noticia/2014/04/10-sinais-para-detectar-bdoencas-cardiacas-em-caesb.html. Acesso em: 28 de ago. 2024

PETMAG. **Doenças cardíacas são responsáveis pela morte precoce de 1 em cada 10 cães e gatos no Brasil.** PetMag, 12 dez. 2019. Disponível em: https://www.petmag.com.br/14636/1-10-animais-estimacao-morrem-causa-problemas-cardiacos. Acesso em: 2 jun. 2024.

REACT. **React:** A biblioteca para web e interfaces de usuário nativas. Disponível em: https://pt-br.react.dev. Acesso em: 21 de abr. 2024

SANTOS, Rullyan Gabriel. **ESP32**. Disponível em: https://deinfo.uepg.br/~alunoso/2019/SO/ESP32/HARDWARE/. Acesso em: 15 de mai. 2024

CUNHA, Andre. **Como instalar e configurar o Expo do React Native**. Disponível em: https://www.alura.com.br/artigos/como-instalar-configurar-expo-do-react-native. Acesso em: 31 maio. 2024.