UNIVERSIDADE FEDERAL DO PARÁ
FACULDADE DE ENGENHARIA DE COMPUTAÇÃO

NATHANNAEL SANTOS DA FONSECA

PROCESSAMENTO DE SINAIS VITAIS DE ANIMAIS DE PEQUENO E GRANDE PORTE: UM ESTUDO DE CASO VOLTADO A IOT

CASTANHAL/PA
2024

NATHANNAEL SANTOS DA FONSECA

PROCESSAMENTO DE SINAIS VITAIS DE ANIMAIS DE PEQUENO E GRANDE PORTE: UM ESTUDO DE CASO VOLTADO A IOT

Trabalho de Conclusão de Curso, apresentado como requisito parcial para a obtenção de grau de Bacharel em Engenharia de Computação, pela Universidade Federal do Pará.

Orientador: Prof. Dr. Bruno Souza Lyra Castro
Universidade Federal do Pará

CASTANHAL/PA
2024

NATHANNAEL SANTOS DA FONSECA

PROCESSAMENTO DE SINAIS VITAIS DE ANIMAIS DE PEQUENO E GRANDE PORTE: UM ESTUDO DE CASO VOLTADO A IOT

Trabalho de Conclusão de Curso, apresentado como requisito parcial para a obtenção de grau de Bacharel em Engenharia de Computação, pela Universidade Federal do Pará.

DATA DE APROVAÇÃO:

CONCEITO:

Prof. Dr. Bruno Souza Lyra Castro
Orientador - UFPA

Membro -

Membro -

CASTANHAL/PA
2024

Dedico aos meus pais e a toda a minha família.

AGRADECIMENTOS

Agradeço primeiramente a Deus, por me conceder a oportunidade e força de cursar Engenharia de Computação na Universidade Federal do Pará. Tudo por Ele e para Ele.

A toda a milha família, desde meus avós João e Eunice Santos, meus bisavós Gildacio e Santana Fonseca(em memória), meus tios e tias, primos e primas, a meus pais, meus irmãos. Vocês foram meu incentivo e apoio, sempre me encorajando a continuar e buscar meus objetivos.

Ao professor doutor Bruno Lyra pela oportunidade, assumindo a responsabilidade de ser meu orientador.

Também ao corpo de docentes da Faculdade de Engenharia de Computação da UFPA campus Castanhal, que me mostraram o caminho, sempre muito aplicados em cada aula. Aos secretários e funcionários da instituição pelo cuidado e atenção.

Aos meus amigos de classe da graduação, a Eng2019.4, pelas muitas conversas e o convívio a cada semestres juntos.

Aos membros dessa bancada.

A minha mãe e pedagoga, Elza, que foi minha primeira professora e incentivadora.

"A tecnologia não é nada. O que é importante é que você tenha fé nas pessoas, que elas são basicamente boas e inteligentes, e se você lhes der as ferramentas certas, elas vão fazer coisas incríveis.' (Steve Jobs)

RESUMO

A IoT (Internet das Coisas) é uma realidade em diversos ambientes, desde a indústria até no seu uso domestico. Essa tecnologia vem mudando e facilitando a vida das pessoas, otimizando tarefas e proporcionando total conectividade entre os mais diversos aparelhos. Aplicar IoT no monitoramento da saúde de animais é um passo importante na integração de tecnologias no cuidado e auxílio de pesquisadores na medicina veterinária. Visando o bem-estar animal, a pesquisa explora desenvolver um sistema de aquisição e transmissão de dados que permita o acompanhamento em tempo real da saúde de animais de pequeno e grande porte, visando a coleta e transmissão de dados vitais como pressão sanguínea e batimentos cardíacos.

Palavras-chave: Internet das Coisas. Saúde animal. Monitoramento em tempo real. Sensor. Filtro de Sinal.

ABSTRACT

IoT (Internet of Things) is a reality in many environments, from industry to domestic use. This technology has been changing and making people's lives easier, optimizing tasks and providing full connectivity between a wide range of devices. Applying IoT to animal health monitoring is an important step in integrating technologies into the care and support of researchers in veterinary medicine. Aiming at animal welfare, the research explores developing a data acquisition and transmission system that allows real-time monitoring of the health of small and large animals, aiming at the collection and transmission of vital data such as blood pressure and heart rate.

Keywords: Internet of Things. Animal health. Real-time monitoring. Sensor. Signal Filter.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

A Internet das Coisas (IoT) tem mudado muitos setores, incluindo a indústria e a saúde. No mundo moderno, sua pesquisa aplicada ao monitoramento da saúde de animais é crucial, melhorando a saúde e o bem-estar dos animais, bem como a indústria pecuária, a medicina veterinária e a conservação da vida selvagem. Essa área de estudo visa fornecer informações úteis sobre como evitar, diagnosticar e tratar doenças em animais.

Cada vez mais, sistemas são pensados com a integração de biossensores para constituir sistemas de monitoramento da saúde animal. Em geral, o quadro de IoT para esse tipo de sistema funciona nas seguintes etapas: coleta, transferência de dados e o banco de dados.

Os sensores extraem os dados de saúde animal, esses dados são coletados por microcontroladores que funcionam como a unidade de transferência, que fica responsável por transmitir esses dados via comunicação sem fio para o data center. No Data center os dados são salvos no banco de dados para análise e visualização.

Tal tipo de tecnologia permite que o cuidado com o animal possa ser feito em qualquer lugar, podendo ser usado nas mais variadas formas, desde monitorar a pressão sanguínea, batimentos cardíacos, frequência respiratória, e outros sinais vitais. O cuidador ou responsável pela saúde animal pode monitorar os status da saúde em tempo real, agilizando tomada de decisões e possibilitando a prevenção de possíveis doenças.

1.1 Objetivos

1.1.1 Objetivo geral

Pretende-se construir um sistema de aquisição e transmissão de dados utilizando Internet das Coisas(IoT), aplicadas ao monitoramento de Saúde de animais de pequeno e grande porte, para o auxílio às pesquisas na área de medicina veterinária no acompanhamento dos animais.

1.1.2 Objetivos específicos

Desenvolver um dispositivo para o monitoramento de saúde animal, onde podemos coletar dados como temperatura corporal, frequência cardíaca e sinais ECG do animal e posteriormente essas dados possam estar acessíveis em plataforma web. O desenvolvimento se dá com base na viabilidade dos equipamentos usados, bem como no conforto do animal e na precisão na coleta e na análise das informações coletadas.

O projeto engloba a criação de um dispositivo vestível e personalizado, além da pesquisa de técnicas de processamento digital de sinais e a transmissão segura desses dados, para realizar o tratamento adequado e a confiável transmissão dos dados coletados pelo dispositivo.

1.2 Estrutura do trabalho

Este trabalho está dividido em cinco seções, referências, anexos e apêndices.

Na seção 1 é apresentado o contexto no qual o trabalho está inserido, a justificativa e os objetivos almejados.

A seção 2 mostra conceitos teóricos relacionados os estudos realizados como as noções de siais médicos e a identificação do que é um sinal. Além das ferramentas utilizadas no estudo tal como ESP8266.

Na seção 3, os resultados são apresentados juntamente com suas devidas discussões, verificando a eficiência do modelo proposto e sua aplicação em mundo real.

Finalizando, a seção 4 faz as devidas conclusões e apresenta sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Introdução

O presente trabalho traz um estudo de como no âmbito acadêmico é possível relacionar tecnologias com o IoT para a criação de um sistema de coleta de dados médicos, que tem o intuito de auxiliar pesquisadores da área de Medicina veterinária no cuidado e prevenção de doenças cardíacas em animais. Antes de apresentar detalhes sobre o sistema proposto é preciso definir alguns conceitos que serão importantes no decorrer deste trabalho.

O termo "Internet das Coisas"(IoT) foi cunhado por Ashton (1999), enquanto trabalhava no MIT(Massachusetts Institute of Technology). Ele utilizou o termo em uma apresentação que tratava do uso de sensores RFID(Radio Frequency Identification) para otimizar a gestão de cadeias de suprimentos. A visão de Ashton era de que futuramente todos os objetos físicos utilizados em nosso cotidiano poderiam estar conectados à internet, permitindo a coleta e o compartilhamento de dados em tempo real. Apesar de cunhado por Aston, a ideia de conexão de dispositivos de forma contínua a internet já vinha sendo amplamente discutida, assim essa ideia de interconexão entre dispositivos e sistemas é a base do que temos hoje como IoT.

O Eletrocardiograma é um exemplo de como a IoT pode ser aplicada de maneira eficaz na coleta de dados médicos. Monitorar as atividades elétricas do coração, fornece informações essências sobre o ritmo e a frequência cardíaca. Em um cenário IoT, sensores conectados ao corpo do paciente podem enviar dado em tempo real para plataformas remotas, permitindo a análise contínua do estado de saúde e a detecção precoce de anormalidades, como arritmias cardíacas. A coleta e o envio automatizado desses sinais para sistemas de análise e bancos de dados, otimizam o acompanhamento médico.

Pensando em um ambiente de monitoramento de animais na área de medicina veterinária, animais de pequeno e grande porte, como cavalos e bovinos, muitas vezes necessitam de monitoramento constante, especialmente em condições críticas. Como proposto por (KARTHICK; SRIDHAR; PANKAJAVALLI, 2020), ao usar sensores que captam sinais de ECG em animais, é possível acompanhar a saúde cardíaca remotamente, algo que pode ser crucial para que pesquisadores estudem padrões cardíacos.

2.2 Eletrocardiograma

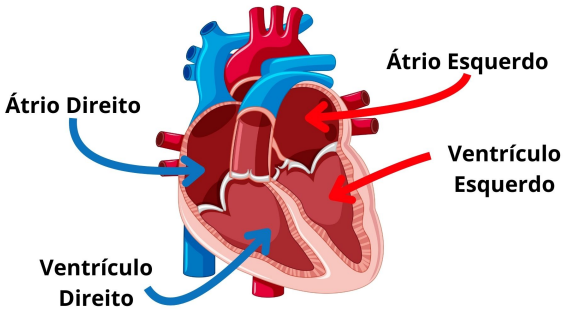
2.2.1 Noções de anatomia e Fisiologia do coração

O coração é um dos órgãos mais vitais para sobrevivência de todos os mamíferos, em geral, sua responsabilidade é bombear o sangue por todo o corpo, garantindo o transporte de nutrientes para os tecidos e remover resíduos metabólicos. O coração tem apresenta a seguinte estrutura, dividida em quatro câmeras: dois átrios e dois ventrículos, que trabalham em conjunto para manter o fluxo de sanguíneo eficiente.

As câmaras são separadas por válvulas que geram o fluxo unidirecional do sangue, sendo a válvula tricúspide(entre o átrio direito e o ventrículo direito) e a válvula mitral(entre o átrio esquerdo e o ventrículo esquerdo), conforme mostrado na Figura 2.1.

O átrio direito recebe o sangue desoxigenado vindo do corpo através das ceias cavas, por meio das válvulas, esse sangue passa para o ventrículo direito que bombeia esse sangue para os pulmões por meio da artéria pulmonar, onde será oxigenado. O átrio esquerdo

Figura 2.1 – Estrutura do Coração



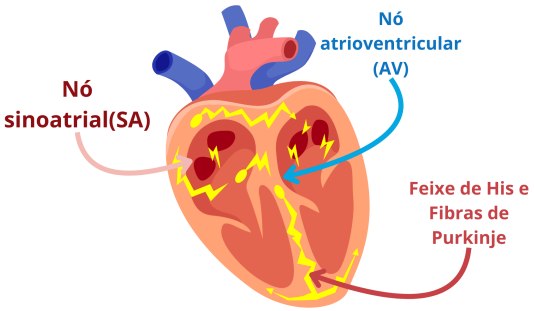
recebe o sangue oxigenado que retorna dos pulmões pelas veias pulmonares, chegando ao ventrículo esquerdo o sangue oxigenado é bombeado para o resto do corpo através da artéria aorta.

O ciclo cardíaco é composto por duas fases principais, a primeira é a sístole, a qual é a fase de contração, onde os ventrículos bombeiam o sangue para os pulmões, no caso do ventrículo direito, ou para o restante do corpo, no caso do ventrículo esquerdo.

A segunda fase é a Diástole, também conhecida como fase de relaxamento, quando os ventrículos se enchem de sangue novamente. Essa é uma fase de preparação onde o coração realiza os processos para a próxima contração.

O coração apresenta um sistema de condução elétrica, que regula o ritmo das contrações do coração com impulsos elétricos, esse sistema é chamado de sistema de condução, que inclui: Nó Sinoatrial(SA), Nó Atrioventricular(AV), Feice de His e Fibras de Purkinje.A Figura 2.2 mostra onde seriam as respectivas localizações de cada um dos componentes nesse sistema.

Figura 2.2 – Fisiologia do coração



O Nó Sinoatrial é conhecido como o “marcapasso natural” coração. Localizado no átrio direito, ele gera os impulsos elétricos, que se propagam pelos átrios causando sua contração, esses impulsos se iniciam a cada batida cardíaca. O Nó Atrioventricular retarda o impulso do nó SA, permitindo a contração dos átrios, que esvaziam completamente o sangue nos ventrículos antes que estes se contraiam. Por fim, o Feixe de His e as Fibras de Purkinje conduzem esses impulsos que chegam aos ventrículos, fazendo com que se contraiam e expulsem o sangue.

Entender a anatomia e a fisiologia do coração foi essencial para seguir os estudos e na montagem desse projeto, pois o eletrocardiograma (ECG) registra essas atividades elétricas do coração, essa é uma ferramenta fundamental para identificar problemas no

ritmo cardíaco. Por meio da coleta de sinais cardíacos é possível detectar irregularidades e agir rapidamente para a saúde do animal.

Em animais, o tamanho, a forma e a funcionalidade do coração pode variar conforme a espécie. Como exemplo, o coração de um cavalo é proporcionalmente maior e mais poderoso, devido à necessidade de bombear grandes volumes de sangue para sustentar sua atividade física intensa. Em contraponto, animais de pequeno porte, com cães e gatos, possuem um coração menor, mas ainda assim com ritmos cardíacos elevados, exigindo um controle mais rápido do fluxo sanguíneo.

Na medicina veterinária, a compreensão da anatomia e fisiologia do coração é crucial para que veterinários possam conduzir intervenções adequadas, não apenas para tratar doenças, mas também para prevenir complicações em espécies que apresentam predisposições genéticas a problemas cardíacos. A contínua coleta de dados e estudos sobre a saúde cardiovascular animal tem o potencial de promover avanços significativos, tanto em termos de diagnóstico quanto de tratamentos menos invasivos e mais eficazes.

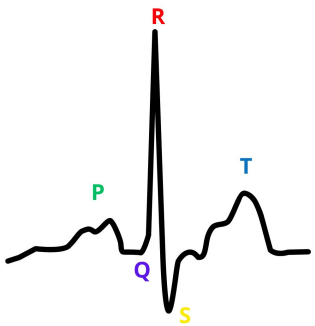
2.2.2 Eletrocardiograma (ECG)

O eletrocardiograma (ECG) é um exame médico que registra a atividade elétrica do coração. Este exame é essencial para o diagnóstico de diversas condições cardíacas, como arritmias, infarto do miocárdio, e anormalidades na condução elétrica do coração. O ECG capta os sinais elétricos gerados pelo coração durante sua contração e relaxamento, permitindo a análise detalhada de como o coração está funcionando.

Como explicado na sessão anterior, o coração gera impulsos elétricos que controlam os batimentos cardíacos. Esses impulsos começam no nó SA e se propagam pelos átrios e ventrículos, provocando sua contração. O eletrocardiograma mede essas variações elétricas por meio de eletrodos colocados em pontos específicos do corpo. Em humanos esses locais são geralmente os braços, pernas e tórax.

Através dos eletrodos, o ECG detecta a diferença de potencial gerada pela despolarização e repolarização do músculo cardíaco. Esses sinais são registrados em um gráfico, exemplificado na Figura 2.3, onde as ondas típicas de um ciclo cardíaco podem ser observadas: Onda P: Representa a despolarização dos átrios; complexo QRS: Corresponde à despolarização dos ventrículos; onda T: Relaciona-se à repolarização ventricular.

Figura 2.3 – Ondas do ciclo cardíaco



É importante frisar que o complexo QRS possui alguns modelos para detecção, em nosso estudo, tentamos aplica o algorítimos de Pam & Tompkins (COSTA; COSTA; REGIS, 2015). Esse algoritmo é amplamente utilizado para a identificação da frequência cardíaca e análise de arritmias, pois o complexo QRS representa a despolarização dos

ventrículos e está associado com o batimento cardíaco. A detecção precisa desse complexo é crucial para monitoramento e análise de saúde cardíaca.

O ECG é amplamente utilizado para o diagnóstico de arritmias(Irregularidades no ritmo cardíaco), infarto do miocárdio(O bloqueio no suprimento sanguíneo ao coração, que pode ser identificado por alterações específicas no traçado do ECG, como elevações no segmento ST) e também bloqueios cardíacos(Problemas na condução elétrica entre as câmaras do coração).

Para um sistema de monitoramento em tempo real de sinais como frequência cardíaca e ritmo cardíaco, a utilização de dispositivos portáteis de ECG usando microcontroladores como o ESP8266 que atuando em conjunto com os sensores de ECG, vem ganhando força em projetos de monitoramento de saúde. Um sistema para monitoramento contínuo desses sinais representa um grande auxílio para pesquisadores na área de medicina veterinária.

2.3 Sinais e Filtros Digitais

2.3.1 Sinais Analógicos

Sinais analógicos são representações contínuas de grandezas físicas. Ao contrário dos sinais digitais, que assumem valores discretos, os sinais analógicos variam de forma contínua ao longo do tempo, podendo assumir infinitos valores em um intervalo. No contexto de medições médicas, como o eletrocardiograma ((QUADROS et al.,)), o sinal analógico representa a variação contínua da atividade elétrica do coração em função do tempo.

Um sinal analógico passa por duas etapas principais para ser processado, primeiramente ele é capturado por um sensor e convertido em um sinal elétrico contínuo. Esse sinal passa por um conversor analógico-digital(ADC), que amostra em intervalos regulares(amostragem) e transforma esses valores contínuos em números discretos(quantização). Uma vez que o sinal é digitalizado, ele pode ser processado por softwares para aplicar filtros, remover ruídos e extrair informações úteis.

2.3.2 Filtros digitais

Após a etapa de conversão do sinal analógico em digital, é comum que esse apresente ruídos, devido à interferência de outros dispositivos eletrônicos, movimentos físicos de sensor e outros. Para garantir que o sinal seja limpo e útil para a análise, são aplicados filtros digitais, que tem o objetivo eliminar a componente indesejada do sinal sem comprometer suas características importantes.

Como no sinal ECG, onde a respiração do paciente, pelos, movimentação do paciente e até mesmo circuitos elétricos do dispositivo causam ruído no sinal, é de grande valor que esse sinal seja filtrado e que as características do sinal cardíaco permaneçam inalteradas para um diagnóstico preciso e a intervenção da equipe de veterinária possa ser correta.

Existem diferentes tipos de filtros digitais, entre os mais comuns temos: Filtros Passa-Baixa, Filtros Passa-Alta, Filtros Passa-Banda e Filtros Notch. Em nossa pesquisa, para aplicar um filtro em um sinal ECG utilizamos o filtro Butterworth, sendo um filtro da categoria passa-baixa. Isso porque, no processamento de Sinais ECG, um filtro passa-baixa Butterworth é frequentemente utilizado para remover ruídos de alta frequência(como a interferência elétricas) enquanto preserva as frequências cardíacas mais baixas, fundamentais para a análise do sinal.

O filtro Butterworth é conhecido por sua resposta de magnitude suave, ou seja, sem ondulações na banda de passagem, e uma atenuação eficiente na banda de rejeição. Para

explicar melhor seu funcionamento é importante entender alguns conceitos.

Conforme estudos disponibilizados pela (Universidade Estadual Paulista (UNESP), 2024), a função de transferência de um filtro digital descreve como o filtro responde a diferentes frequências. No domínio Z, ela é dada pela razão entre os polinômios dos coeficientes de saiada e os coeficientes de entrada. Isso é feito a partir da equação:

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_N z^{-N}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_M z^{-M}}$$

- **Numerador(coeficientes b):** determina a distribuição dos sinais de entrada
- **Denominador(com coeficientes a):** determina a contribuição dos sinais de saída.

Esses coeficientes são calculados para satisfazer os requisitos de atenuação e resposta em frequência do filtro. Para isso se dever ter uma base que se utiliza a Ordem do filtro, que define o quão afiado é o corte do filtro, ou seja, se o filtro fará uma transição mais bruta entre a banda de passagem e a banda passante.

A frequência de corte, na qual definimos a frequência na qual o filtro começa a atenuar o sinal e por fim devemos definir o tipo de filtro. Com essa base, podemos projetar um filtro Butterworth, garantindo uma resposta de magnitude plana.

Com isso, podemos definir matematicamente a resposta de frequência do Butterworth, dada por:

$$|H(j\omega)| = \frac{1}{\sqrt{1 + \left(\frac{\omega}{\omega_c}\right)^{2n}}}$$

- ω: frequência angular.
- ω_c: frequência de corte.
- **n**: Ordem do filtro.

Para a implementação do filtro nesse projeto, utilizamos a ferramenta **SciPy** em Python, para calcular os coeficientes automaticamente e após aplicamos esses coeficientes. Um modo de aplicar esses coeficientes é utilizar a fórmula da diferença, dada por:

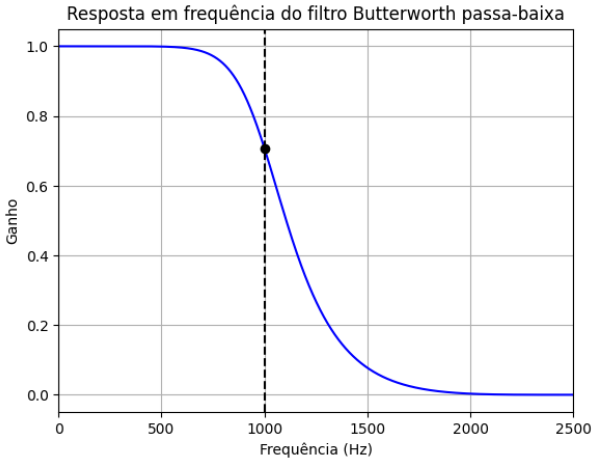
$$y[n] = b_0 x[n] + b_1 x[n-1] + \cdots + b_N x[n-N] - a_1 y[n-1] - \cdots - a_M y[n-M]$$

Contudo, a própria biblioteca oferece funções que viabilizam o uso automático desse calculo, algo que será melhor abordado nas próximas sessões.

A Figura 2.4 mostra a resposta em frequência de um filtro Butterworth passa-baixa. Nesse exemplo o filtro foi definido com ordem 4 com uma frequência de corte a 1000Hz e frequência de amostragem a 5000Hz. No gráfico, o eixo horizontal representa a frequência em hertz (Hz) e o eixo vertical representa o ganho, ou seja, o quanto o sinal é preservado ou atenuado em diferentes frequências. O filtro Butterworth é caracterizado por uma transição suave da banda passante (frequências preservadas) para a banda atenuada (frequências filtradas).

No gráfico, o ganho é próximo de 1 (ou seja, o sinal passa sem ser atenuado) até a frequência de corte, marcada pela linha pontilhada preta. Após essa frequência (aproximadamente 1000 Hz), o filtro começa a atenuar as frequências gradualmente, de forma que em frequências muito mais altas, o ganho se aproxima de 0, eliminando os componentes de alta frequência. Isso torna o filtro ideal para remover ruídos de alta frequência, preservando o sinal de interesse na faixa mais baixa, como em sinais de ECG.

Figura 2.4 – resposta em frequência



2.4 Componentes(Hardware e Software)

2.4.1 Microcontrolador: NodeMcu

O NodeMCU é uma plataforma de desenvolvimento baseado no microcontrolador ESP8266, amplamente utilizado em projetos de IoT. Amplamente utilizado em trabalhos como em (CUNHA, 2012) Ele integra um módulo WI-Fi, facilitando a conexão com redes sem fio e oferece uma interface de fácil uso para programadores, sendo compatível com a linguagem de programação Lua ou o ambiente de desenvolvimento Arduino IDE.

Figura 2.5 – NodeMcu

O microcontrolador é construído em torno do ESP8266, sendo altamente versátil e de baixo custo. Dentre as suas principais características podemos destacar o Wi-fi integrado, baixo consumo de energia, oferece pinos de entrada e saída digitais e um pino analógico que permite a integração com sensores, além de compatibilidade com Arduino IDE.

2.4.2 Sensor de Batimentos Cardíacos: AD8232

O AD8232 é um sensor especializado na captação de sinais de eletrocardiograma (ECG), projetado para monitorar a atividade elétrica do coração. Ele é responsável por captar os sinais cardíacos do usuário e convertê-los em dados analógicos, o qual é então

transmitidos para o ESP8266. Este sensor é ideal para aplicações em monitoramento de saúde, oferecendo precisão e baixo consumo de energia.

2.4.3 Plataforma de Desenvolvimento: Arduino IDE

O Arduino IDE é um ambiente de desenvolvimento utilizado para programar o ESP8266. Ele fornece uma interface simples e intuitiva para escrever, compilar e carregar o código no microcontrolador.

Figura 2.6 – ArduinoIDE

O uso do Arduino IDE facilita o desenvolvimento de projetos com ESP8266, permitindo o uso de diversas bibliotecas que ajudam a simplificar a comunicação com sensores e outros componentes.

2.4.4 Protocolo de Comunicação e Broker MQTT(Message Queuing Telemetry Transport)

O protocolo MQTT é utilizado para a transmissão de dados entre o ESP8266 e o servidor de armazenamento. Ele é um protocolo leve e de uso já aplicado em outros estudos como em (ROCHA, 2018), ideal para aplicações IoT, onde a largura de banda e o consumo de energia são limitados. No sistema, o MQTT é responsável por enviar os dados captados pelo ESP8266 para o broker MQTT, que age como um servidor intermediário, esse envio funciona com um sistema de *publish/subscribe*.

O Broker MQTT é um servidor que gerencia a comunicação entre dispositivos e aplicações. Ele recebe os dados enviados pelo ESP8266 via MQTT e os encaminha para os sistemas de processamento e armazenamento. No sistema, o broker atua como o intermediário responsável por organizar e distribuir as mensagens para os destinos apropriados, como bancos de dados ou interfaces gráficas.

O *client publisher* é o dispositivo que envia(publica) o dado que deseja ser enviado para um tópico específico no *broker* configurado. Já o *client subscribe* faz a inscrição em um ou mais tópicos para receber todas a mensagens publicas nesse tópico.

Assim, quando o *broker* recebe uma mensagem publicada em um tópico, ele distribui essa mensagem para todos os *client subscribers* inscritos no tópico correspondente. O *broker* conhece tanto ambos os clients e atua como ”ponte”direcionando cada publicação.

Esses tópicos possuem uma estrutura hierárquica, que se assemelha a um caminho de arquivos, não é necessário realizar nenhuma configuração previa, apenas o cliente se conectar ao *ip* do *broker* e publicar já é suficiente. Como exemplo de tópico temos:

$$brokerMosquitto/sinalEcg$$

Juntamente dos tópicos é possível utilizar *willdcards*, que atuam para facilitar a obtenção de mensagens de um ou mais tópicos. É possível utilizar dois tipos de *willdcards*, + funciona como um curinga que representa um único nível na hierarquia do tópico. Por exemplo, o tópico sensor/+/temperatura/+ capturaria dados de temperatura de diferentes sensores e dispositivos em vários locais. Já o caractere # representa todos os níveis subsequentes e deve ser usado apenas no final da assinatura do tópico. Assim, uma assinatura como a/b/# receberia mensagens de todos os subníveis, como a/b/c/d, a/b/c, etc.

Além disso, o protocolo MQTT oferece níveis de QoS(Qualidade de Serviço) que controlam a entrega das mensagens, fundamentalmente temos 3 níveis de QoS. O QoS 0 (Entrega pelo menos uma vez) é o nível mais simples, onde a mensagem é enviada apenas uma vez, sem confirmação. Nesse caso, se a mensagem for perdida durante o envio, ela não será retransmitida.

No QoS 1 (Entrega pelo menos uma vez), a mensagem é confirmada pelo broker, e se o publisher não receber essa confirmação, ele reenviará a mensagem. Já o QoS 2 (Entrega exatamente uma vez) garante a entrega da mensagem apenas uma vez, utilizando um protocolo de confirmação mais complexo para evitar duplicatas, sendo o nível mais robusto e seguro.

O protocolo MQTT oferece vantagens para o seu uso nesse projeto, como a sua leveza e eficiência, o que é ideal para dispositivos com baixa capacidade de processamento.Em suma, é um protocolo que suporta diversos dispositivos e conexões ao mesmo tempo, demonstrando sua escalabilidade. É possível obter mais informações sobre o protocolo no *website* da HiveMQ, onde temos uma descrição do MQTT como um protocolo leve e eficiente para IoT (HIVEMQ, 2024).

2.4.5 Node-RED

O Node-RED, (NODE-RED, 2024), é uma aplicação baseadas em fluxos, com base no Node.js, projetada para conectar dispositivos, *APIs* e serviços online de maneira visual. A ferramenta possui um editor que pode ser acessado em um navegador web, onde é possível criar e configurar fluxos de dados usando blocos, conhecidos como nós.

Uma das razões para a escolha da ferramenta para essa aplicação é sua característica escalável, podendo ser usada em diversas plataformas, desde um pequeno dispositivo com o Raspberry Pi até servidores mais robustos. Operando em um modelo de fluxo assíncrono, o processamento de mensagens é eficiente.

A interface visual do Node-RED, Figura 2.7, é dividida em três partes principais. O Painel de Nós, localizado à esquerda, exibe uma lista de nós pré-configurados, organizados por categorias como **Entrada, Saída, Função e Análise**. No Editor de Fluxos, no centro, você pode arrastar e conectar os nós para criar e construir seus fluxos. À direita, a Janela de Propriedades permite configurar as propriedades dos nós e visualizar a saída de *debug*.

Os fluxos no Node-RED são executados assim que uma mensagem é recebida por um nó de entrada. Os fluxos são *event-driven*, ou seja, a execução depende da chegada de uma mensagem. Cada nó processa a mensagem e a repassa para o próximo nó conectado. O fluxo termina quando a mensagem chega a um nó que não tem saída (como o nó *Debug*).

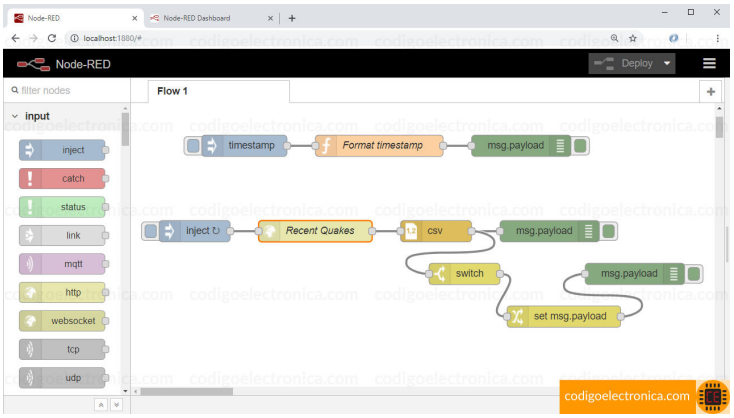


Figura 2.7 – Painel Node-Red

As mensagens em Node-RED são objetos JavaScript que seguem uma estrutura padronizada. O campo mais importante é o *msg.payload*, que contém os dados principais a serem processados.

Outro ponto de interesse para a utilização do Node-RED é sua fácil integração com dispositivos de hardware, serviços na nuvem e APIs por meio de métodos como MQTT, que permite comunicação com dispositivos IoT, *WebSocket*, para comunicação bidirecional em tempo real, e integração com bancos de dados, utilizando nós específicos para SQL, MongoDB, redis, entre outros.

2.4.6 Software de Processamento e Armazenamento

Python: O Python é utilizado para o tratamento dos dados recebidos. Ele processa os sinais brutos do sensor, como filtragem de ruído, detecção de picos e cálculo de métricas relevantes, como frequência cardíaca. Fazemos esses cálculos utilizando as bibliotecas Numpy e SciPy, principalmente, muito utilizadas para o uso de dados matemáticos.Python também é usado para automatizar o envio dos dados para o banco de dados.

MySQL: O MySQL é o sistema de gerenciamento de banco de dados relacional utilizado para armazenar os dados de batimentos cardíacos de forma estruturada. Ele permite consultas eficientes e a manipulação dos dados para análises posteriores. Desenvolvimento *Web*

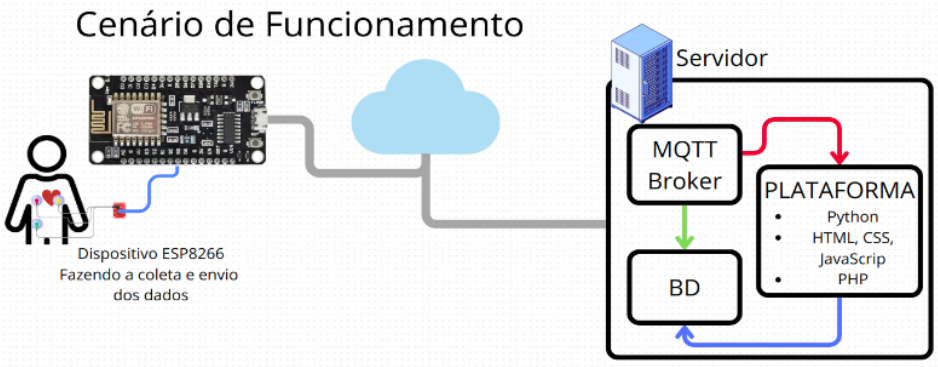
HTML, CSS, JavaScript, PHP: Essas tecnologias são utilizadas para desenvolver a interface web do sistema, onde os usuários podem visualizar os dados de batimentos cardíacos em tempo real. O HTML define a estrutura da página, o CSS estiliza a interface, e o JavaScript permite a interação dinâmica com os dados. O PHP é utilizado no *backend* para interagir com o banco de dados e fornecer os dados ao *front-end*.

3 METODOLOGIA

3.1 Arquitetura

Para o desenvolvimento eficiente de um dispositivo para monitoramento da saúde animal, foi implementado um protótipo para a coleta de dados de eletrocardiograma. Os sinais coletados são transmitidos para o servidor central, o sinal é carregado na plataforma web onde é possível fazer a visualização em tempo real, após um tratamento inicia para remoção de ruído, finalmente os dados são analisados e armazenados. Conforme a Figura 1, seguimos essa arquitetura para o desenvolvimento do projeto.

Figura 3.1 – Arquitetura



Durante o processo de desenvolvimento, também utilizamos um sensor de temperatura como um dos recursos auxiliar para os testes preliminares, antes mesmo de adquirirmos o sensor de batimentos cardíacos. No prosseguimento deste trabalho tivemos a aplicação em duas frentes: no desenvolvimento do Hardware e no desenvolvimento do software.

No desenvolvimento de Hardware se direcionou na estruturação da montagem do ESP8266 com os sensores, bem como a definição dos pinos que foram utilizados, visando o uso adequado de cada tecnologia. Igualmente, visamos fazer o uso pertinente da energia para alimentação dos dispositivos, evitando gasto desnecessários da fonte de alimentação e mantendo a eficiência energética para o pleno funcionamento dos dispositivos utilizados.

No desenvolvimento de software foi organizado para o uso das tecnologias que mais se adequavam para o tipo de dados que precisamos trabalhar e do mesmo modo que fossem compatíveis entre si.

Na Figura 3.1 temos o cenário de funcionamento do modelo proposto, no lado esquerdo temos o microcontrolador ESP8266 conectado ao sensor de batimentos cardíacos(responsável pela coletar os dados do paciente). A nuvem representa a comunicação pela internet, em nosso caso, utilizamos a rede Wi-Fi que conecta o dispositivo ESP8266 ao servidor remoto. No lado direito temos o Servidor, onde hospedamos o MQTT *Broker*, o banco de dados utilizado e plataforma, que processam os dados e exibem para o usuário.

Em resumo, o sistema funciona capturando dados de batimentos cardíacos com o ESP8266, enviando-os via Wi-Fi para um broker MQTT em um servidor. Os dados são então armazenados em um banco de dados e processados por uma plataforma desenvolvida com as tecnologias:Python, HTML, CSS, JavaScript, PHP. Permitindo a visualização e análise dos sinais cardíacos.

Nessa etapa utilizamos os seguintes equipamentos:

- Microcontrolador: ESP8266.
- Sensores: Sensor temperatura NTC10k, Sensor Batimentos Cardíacos Ad8232.
- Plataforma de Desenvolvimento: Arduino IDE.
- Protocolo de Comunicação: MQTT (Message Queuing Telemetry Transport).
- Servidor: Broker MQTT.
- Software: Python para tratamento de dados, MySQL para armazenamento, Node-RED e HTML, CSS, JavaScript, PHP para desenvolvimento do website.

3.2 Procedimento

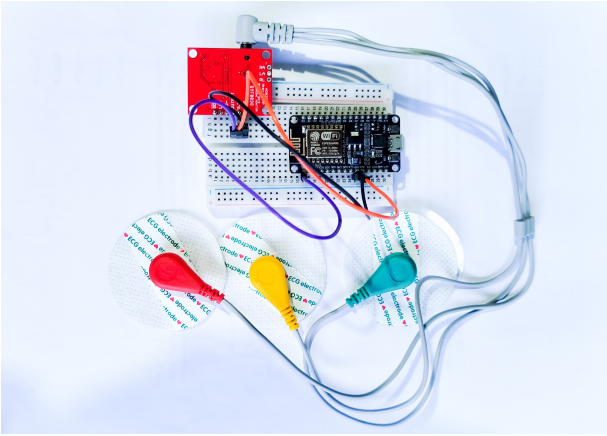
Na fase inicial da montagem do hardware, foi realizada com os sensores de temperatura NTC 10K. O sensor foi escolhido devido à sua versatilidade e precisão, variando de 1% a 5%, com uma faixa de operação de 55 C a 125 C.A interface é simples, usando um pino analógico, facilitando a integração com o ESP8266 por meio de bibliotecas livres e de código aberto.

Além disso, o módulo de batimentos cardíacos que cogitamos utilizar, também usa apenas um pino analógico, fazendo com que a lógica de programação entre os 2 módulos seja parecida. Realizada a conexão do sensor com o microcontrolador ESP8266, NodeMCU, que fica conectado a porta analógica A0 e a partir de então sua programação foi feita na linguagem C/C + + na IDE do Arduino. Os dados coletados pelo ESP são enviados para o broker, esse envio é feito a cada ciclo definido na programação.

Realizada esta etapa de teste de conectividade e envio de dados de temperatura, cogitamos formas para os testes de envio em tempo real de dados ECG. Utilizando os arquivos de dados fornecidos pela plataforma PhysioZoo (SHEMLA; BEHAR, 2019), que disponibiliza amostra de dados coletados de animais, como camundongos, cachorros e coelhos.

Dispondo desses dados, um arquivo foi carregado no ESP8266 com uma parcela dessas amostras, enviadas simulando uma coleta e envio em tempo real dos dados para o broker. Assim podemos tirar informações importantes que auxiliaram na configuração de lógica de envio de informações, bem como a quantidade máxima de buffer que podia ser usada, sem perda de dados.

Figura 3.2 – Dispositivo montado em protoboard



Mediante a aquisição do módulo Sensor Batimentos Cardíacos Ad8232, seguimos a integração com o ESP8266. Utilizando a alimentação na porta 3.3v, conforme especificação

do fabricante, o módulo foi conectado a porta analógica A0 do microcontrolador.

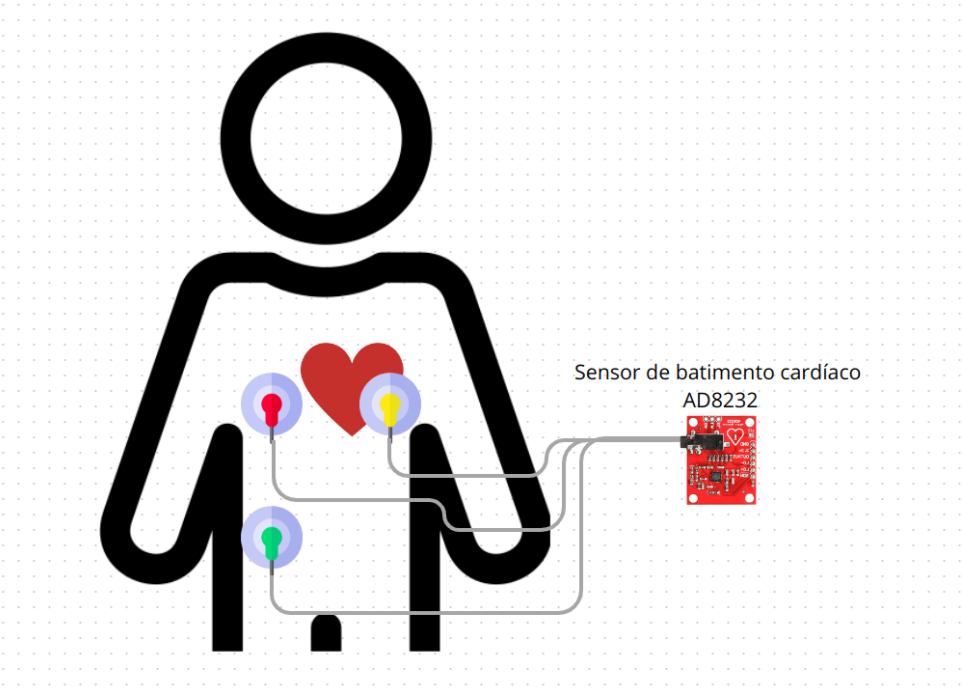
O módulo é um sensor de baixo consumo de energia, contribuindo para a economia de bateria no sistema de dispositivo vestível, apresentando alta precisão na coleta de dados, mantendo a qualidade do sinal com pouco ruído.

Na imagem Figura 3.2 mostra um protótipo de monitoramento de sinais de ECG utilizando um conjunto de componentes eletrônicos montados em uma *protoboard*(placa de ensaio).

Os eletrodos captam o sinal elétrico dos batimentos cardíacos do usuário e o enviam para o sensor AD8232, que amplifica esse sinal e o passa para o NodeMCU. O NodeMCU processa o sinal recebido e o envia via Wi-Fi para um sistema externo (como um servidor com um broker MQTT, mostrado na imagem anterior) para armazenamento ou análise.

Cada eletrodo é encaixado nas almofadas do sensor antes da aplicação no corpo, os eletrodos vem com a seguinte legenda: vermelho RA(right Arm; braço direito), amarelo LA(left arm; braço esquerdo) e verde RL(right leg; perna direita). Utilizamos a seguinte configuração de encaixe das almofadas para a coleta dos dados ECG, conforme a Figura 3.3.

Figura 3.3 – Modelo de representação da utilização do sensor de eletrocardiogramas

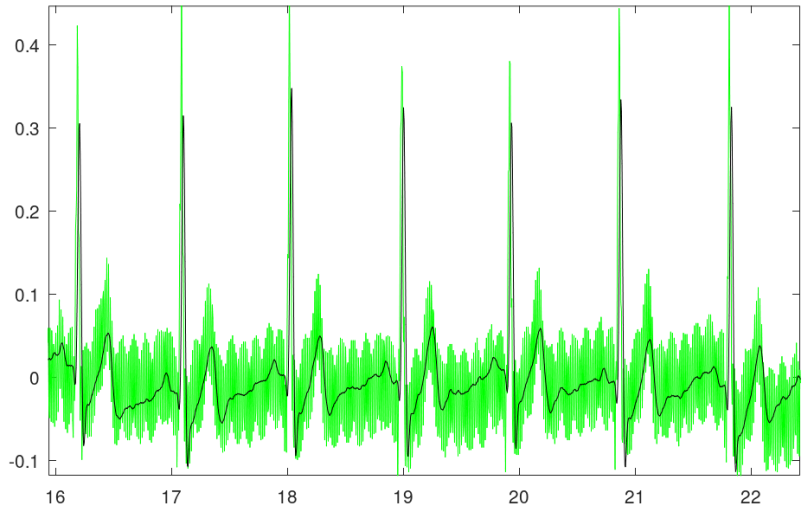


Os primeiros testes realizados, foram conduzidos em um indivíduo, visando avaliar sua funcionalidade e verificar possíveis defeitos de fabricação. A leitura do sensor foi adquirida por meio de uma conexão com a porta analógica do ESP, salvamos os dados da leitura por meio da conexão serial do microcontrolador com o notebook. A Figura 3.4 apresenta o gráfico resultante da leitura obtida pelo sensor durante o teste.

Com os testes, coletamos informações importantes do seu funcionamento. Dessa forma, pudemos avaliar a capacidade do dispositivo no fornecimento dos dados de batimento cardíaco, mostrando os dados necessários de modo preciso e confiáveis.

.

Figura 3.4 – Amostra de dados coletados pela leitura do sensor



3.2.1 Microcontrolador - Configurações e teste

O desenvolvimento do sistema embarcado se deu utilizando o ambiente de desenvolvimento do Arduino, o Arduino IDE, onde realizamos a programação do código para o microcontrolador e realizamos as configurações para que pudesse estar compatível com o ESP8266. O software desenvolvido realiza a conexão com o broker MQTT, verifica se o dispositivo permanece conectado para garantir o envio dos dados, faz a captura dos dados conforme o tempo previamente definido entre cada coleta, bem como o envio em bloco dos dados para o servidor central.

Para o uso da IDE do Arduino como ambiente de desenvolvimento para o ESP8266, é necessário realizar alguns ajustes na configuração da plataforma, garantindo a compatibilidade do microcontrolador com o ambiente de desenvolvimento. Esses ajustes envolvem a adição de bibliotecas específicas e a modificação de parâmetros no gerenciador de placas.

Antes de iniciar o desenvolvimento, é preciso adicionar o suporte ao ESP8266 na IDE do Arduino. Para isso, deve-se acessar as configurações da IDE e incluir a URL correspondente ao gerenciador de placas ESP8266. O caminho é: Arquivo - Preferências - URL Adicional para Gerenciadores de Placas. A URL que deve ser inserida é https://arduino.esp8266.com/stable/package_esp8266com_index.json

Com a URL adicionada, é possível instalar o pacote de suporte para o ESP8266 através do menu: Ferramentas > Placa > Gerenciador de Placas. Dentro do gerenciador, deve-se buscar por “ESP8266” e instalar a última versão do pacote disponível.

Após a instalação, o próximo passo é selecionar o modelo correto do ESP8266 que está sendo utilizado. Isso pode ser feito no menu Ferramentas > Placa, onde a placa “NodeMCU 1.0 (ESP-12E Module)” ou outro modelo compatível deve ser escolhido, conforme o hardware utilizado.

Desenvolvimento com ESP8266 requer frequentemente o uso de bibliotecas adicionais, como a ESP8266WiFi para gerenciar a conectividade Wi-Fi, e a *PubSubClient* para o protocolo MQTT, se houver necessidade de comunicação via IoT.

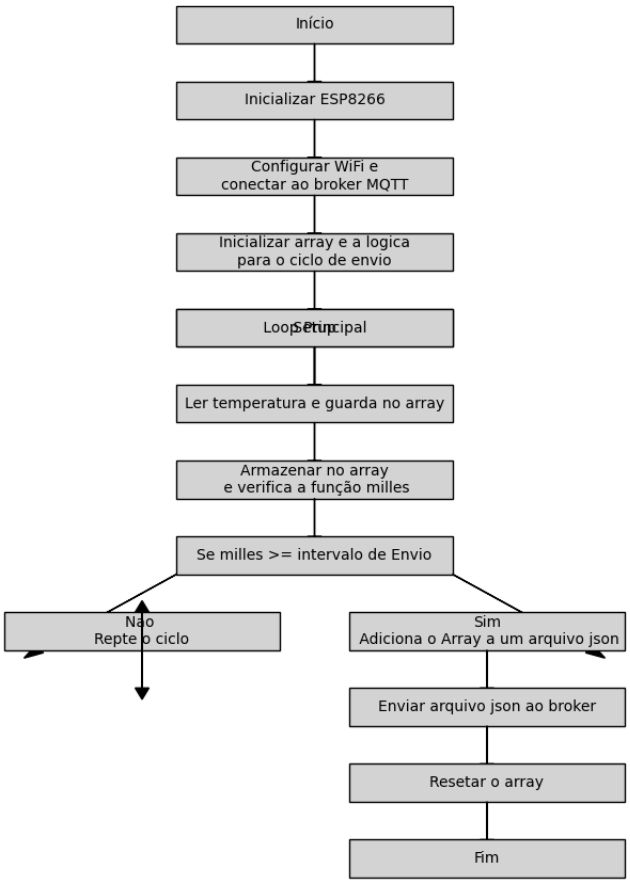
Esses ajustes são essenciais para garantir que a IDE do Arduino reconheça e programe corretamente o ESP8266, permitindo o desenvolvimento de projetos com conectividade à internet e suporte ao protocolo MQTT, como o sistema de monitoramento de

sinais vitais desenvolvido neste trabalho.

Feitos os ajustes configuramos o microcontrolador seguindo o diagrama de blocos conforme na Figura 3.5. Utilizamos algumas bibliotecas integradas ao Arduíno para estabelecer uma conexão com a rede Wi-Fi e com o broker. Os dados são coletados a cada ciclo, sendo armazenados em um *array* para o envio de 25 amostras por vez. Após o envio o *array* é zerado e o ciclo de coleta e envio é iniciado novamente.

A programa no microcontrolador funciona em ciclos, onde dizemos que a função principal é um loop, que se repete a cada ciclo de máquina, sendo o conjunto de operações básicas que um processador realiza para executar uma única instrução de um programa. Ele consiste em buscar a instrução na memória, decodificar a instrução, buscar os operandos, executar a instrução e armazenar o resultado de volta na memória, se necessário. Nesse caso, cada ciclo realizamos uma coleta do sinal, salvo em um arranjo de dados, quando atingimos 25 amostras coletadas o código envia essas informações para o servidor, reiniciando novamente o ciclo até novamente atingir as 25 amostras.

Figura 3.5 – Fluxograma representando a lógica de funcionamento do sistema.



A programação é dividida em 4 partes, o código inicia com a configuração do microcontrolador ESP8266. Primeiramente, são incluídas as bibliotecas necessárias para:

- Gerenciamento da conexão Wi-Fi: A biblioteca ESP8266WiFi.h é utilizada para conectar o dispositivo à rede local, necessária para enviar dados via MQTT.
- Protocolo MQTT: A biblioteca PubSubClient.h é usada para implementar o protocolo MQTT, permitindo que o ESP8266 se comunique com o broker MQTT.

.

Logo em seguida, ocorre a configuração de variáveis de rede, como SSID e senha da rede Wi-Fi, além da configuração do endereço do broker MQTT. A coleta de dados é realizada utilizando a função analogRead() na porta A0 do microcontrolador, que converte os sinais analógicos do sensor de ECG em valores digitais que podem ser manipulados pelo microcontrolador.

Uma vez que os dados são coletados, eles são preparados em um formato adequado para envio via MQTT, geralmente como uma string JSON. Esse formato estruturado permite que os dados sejam interpretados corretamente pelo broker e por outros dispositivos conectados.

O código envia pacotes de dados a cada intervalo de tempo predefinido. Isso é feito para evitar o acúmulo de dados no *buffer*, mantendo a transmissão contínua e evitando perdas de informação. A função *client.publish()* do protocolo MQTT é responsável por publicar os dados no tópico específico do servidor MQTT.

O código também inclui rotinas de tratamento de erros para lidar com possíveis falhas na conexão Wi-Fi ou no broker MQTT. Quando uma desconexão é detectada, o sistema tenta se reconectar automaticamente, garantindo a robustez do sistema. Esse processo é fundamental em um sistema de monitoramento contínuo por garantir que os dados críticos do ECG não sejam perdidos em caso de interrupções momentâneas na rede.

3.2.2 MQTT e NodRed

Para implementar a comunicação via protocolo MQTT no sistema de monitoramento de sinais vitais, foi configurado um *broker* MQTT local utilizando o software *Mosquitto*. O *Mosquitto* é um *broker* amplamente utilizado e compatível com o protocolo MQTT, fornecendo uma solução leve e eficiente para comunicação em sistemas de IoT.

A configuração do *broker* foi baseada nas diretrizes da documentação oficial do Mosquitto. Para garantir a segurança da comunicação, foi definida uma credencial de acesso com usuário e senha, permitindo o controle do acesso ao serviço de mensagens. Adicionalmente, foi configurado o tópico de publicação e assinatura onde os dados de ECG serão transmitidos e coletados.

Um simples teste de publicação de uma mensagem é feita da seguinte forma: mosquitto_pub -h localhost -t test/topic -m ”Mensagem de teste-u ”usuario-P ”senha”

Esses tópicos funcionam como canais de comunicação no qual o ESP8266 publica os dados e o sistema de tratamento recebe e processa as informações. A separação em tópicos específicos permite uma organização adequada dos fluxos de dados, facilitando a gestão e a distribuição das informações no ambiente de IoT.

Após a transmissão dos dados de ECG para o broker MQTT, os mesmos são encaminhados para o Node-RED, uma ferramenta gráfica baseada em fluxo que facilita a integração de dispositivos e serviços. O Node-RED, por sua vez, é responsável por capturar os dados brutos e encaminhá-los para um banco de dados, onde serão armazenados para posterior análise.

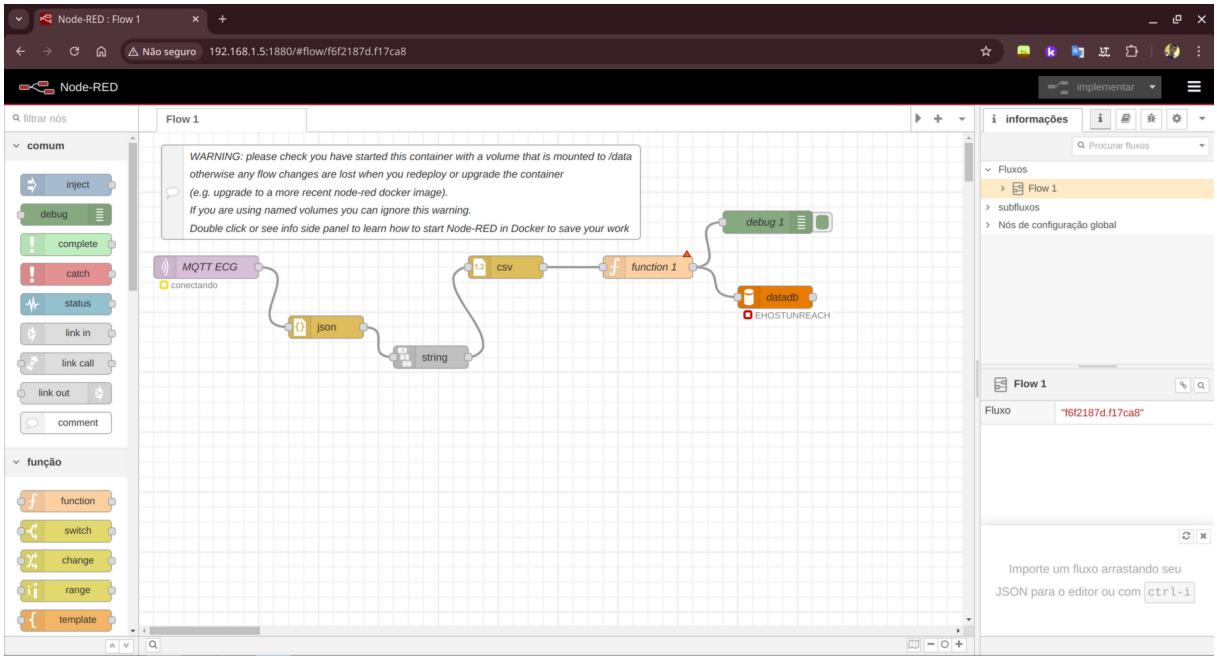
Para integrar o MQTT ao Node-RED é necessário configurar o nó correspondente. A configuração pode ser feita através do menu no Node-RED, selecionando a opção de adição de novos nós.

Com o Node-RED também fazemos a integração com o banco de dados MySQL, fazendo a configuração de leitura e escrita de conexão do Node com o banco especifico.

.

Assim temos o seguinte Fluxo de Trabalho:

Figura 3.6 – Painel do Node-Red



- Configuração do Broker MQTT:
  - Definir o endereço e a porta do broker.
  - Configurar a conexão sem TLS inicialmente.
- Instalação dos Nós no Node-RED:
  - Acessar o menu de instalação e adicionar os nós necessários para MQTT e MySQL.
- Publicação e Recepção de Dados:
  - Utilizar o Node-RED para receber mensagens do broker.
  - Monitorar as mensagens publicadas através da interface do Node-RED.
- Armazenamento em Banco de Dados:
  - Configurar a conexão com o MySQL.
  - Criar fluxos para inserir dados recebidos no banco de dados.

A combinação do broker MQTT local com o Node-RED permitiu a criação de um fluxo eficiente e seguro de coleta, armazenamento e tratamento dos dados de ECG. O uso de tópicos MQTT organizados e a filtragem adequada dos dados garantem a integridade do sistema e a qualidade dos dados para futuras análises clínicas.

3.2.3 Filtragem do sinal

Nesse projeto o filtro utilizado no processamento do sinal de ECG é um filtro Butterworth, a qual é um tipo de filtro passa-banda. Ele é projetado para permitir a passagem de frequências em um intervalo específico, enquanto atenua as frequências fora desse intervalo.

O filtro Butterworth, introduzido por Stephen Butterworth em 1930, é conhecido por sua resposta plana na banda de passagem e uma atenuação progressiva fora da banda de corte. Em um sinal de ECG, os componentes de alta frequência, como ruídos de interferência elétrica e oscilações rápidas, podem mascarar os eventos importantes do ciclo cardíaco, como os picos P, QRS e T. Além disso, componentes de baixas frequências, como

as tendências ou deriva de linha de base, também podem interferir na análise.

Para a construção desse filtro, utilizamos a biblioteca SciPy, uma poderosa ferramenta em Python voltada para computação científica e técnica. A SciPy oferece uma ampla gama de funcionalidades, como integração, otimização, álgebra linear, estatística e, principalmente, processamento de sinais, sendo o foco principal do uso nesta etapa do projeto.

Ao incluir o módulo scipy.signal no código, temos acesso a um conjunto amplo de funções que permitem analisar e manipular sinais. Com a biblioteca e o método necessários importados, o próximo passo é ajustar e verificar os principais parâmetros que serão utilizados no processo de filtragem do sinal.

A transformada de *Fourier* é aplicada inicialmente para identificar e filtrar essas frequências indesejadas. Ao aplicar a FFT, as frequências indesejadas são identificadas e removidas, garantindo um sinal mais limpo para análise clínica.

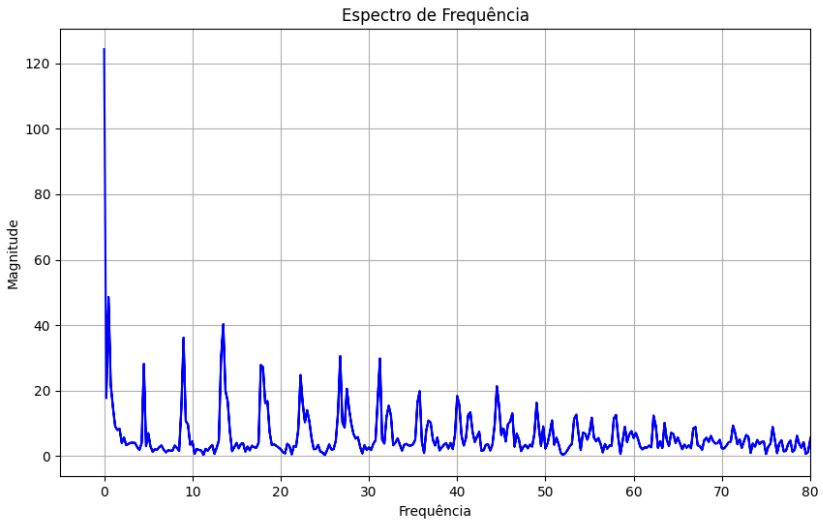


Figura 3.7 – Espectro de Frequência do sinal - Dados obtidos pela plataforma Pysiozoo

Na Figura 3.7, é apresentado o espectro de frequência do sinal de ECG, onde o eixo x representa a frequência (em Hz) e o eixo y, a magnitude do sinal. Observa-se que, a partir de 35 Hz, há uma queda na magnitude da frequência, indicando a presença de ruído, sendo uma componente indesejada do sinal. Com base nisso, definimos a frequência de corte superior em 35 Hz. No início do gráfico, também é visível um pico de magnitude, correspondente ao momento em que o dispositivo foi ligado e iniciou a leitura do sinal de ECG. Essa componente, igualmente indesejada, levou à definição de uma frequência de corte inferior em 0,25 Hz.

Definida essas componentes, passamos para a implementação do filtro Butterworth propriamente dito. O método Butterworth é implementado pela função `butter` (SciPy Developers, 2024a), enquanto a filtragem de sinal é feita com `filtfilt` (SciPy Developers, 2024b).

Utilizamos o método signal.butter, usada para projetar filtros digitais e analógicos do tipo Butterworth. Essa função retorna os coeficientes necessários para implementar o filtro.

Essas frequências de corte escolhida devem ser normalizadas, no caso temos fmim e fmax normalizadas. Para isso utilizamos a frequência de Nyquist(fs2) é definida como

metade da frequência de amostragem (fs). Essa frequência é crucial porque determina o limite superior para a frequência que pode ser adequadamente representada no sinal amostrado. Frequências acima da frequência de Nyquist podem causar *aliasing*, resultando em distorções no sinal.

Essas normalizações são necessárias para as frequências estarem no intervalo [0, 1], sendo o formato esperado pelo filtro Butterworth. Isso garante que o filtro opere corretamente nos limites definidos pela frequência de Nyquist.

Definimos os seguintes parâmetros para o filtro:

- **ordem(int)**: Define a ordem do filtro.
- **[fmin, fmax](array_like):** Define a frequência critica do filtro.
- **btype(string):** Define o tipo de filtro desejado. Valores possiveis são: *'lowpass', 'highpass', 'bandpass', 'bandstop'*.

A ordem do filtro (neste projeto utilizamos um filtro de ordem 2) determina a suavidade da transição entre as frequências passadas e as atenuadas. Um filtro de ordem mais alta terá uma transição mais abrupta, enquanto uma ordem mais baixa resultará em uma transição mais suave. A escolha da ordem 2 é um compromisso que geralmente oferece um bom desempenho em aplicações de filtragem de sinais, permitindo uma boa atenuação das frequências indesejadas sem distorcer excessivamente o sinal.

As faixas de frequência crítica recebem um array contendo as frequências mínima e máxima normalizadas, indicando os limites da banda. Definimos com *'bandpass'* o tipo de filtro.

Com esses parâmetros a função nos retorna os coeficientes *b, a* são os coeficientes do filtro Butterworth(denominador do filtro IIR), *b* representa os coeficientes do numerador e *a* os do denominador da fórmula do filtro, como explicado na seção 2.3.2. A partir disso, podemos fazer a filmagem do sinal, utilizando o método signal.filtfilt.

A função aplica um filtro digital para frente e para trás de um sinal, o que significa que ele aplica um filtro digital linear duas vezes, para frente e para trás. Para o uso dessa função passamos os parâmetros *b, a* calculados na linha anterior, e o *ecg_sinal* é o sinal ECG recebido.

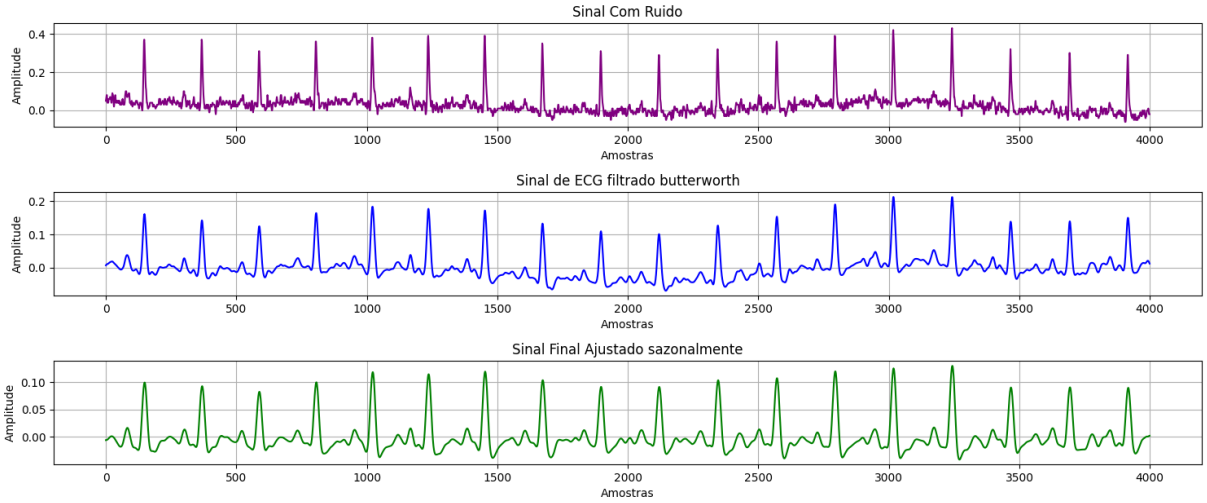
Temos como resultado o *ecg_filt1*, esse é o sinal ECG recebido após a aplicação do filtro. Essencialmente, este trecho de código cria e aplica um filtro passa-banda para extrair apenas as frequências de interesse do sinal de ECG, resultando em um sinal filtrado (ecg_filt1) adequado para análise ou processamento posterior.

Conforme discutido em (GUSTAFFSON, 1996), o artigo apresenta uma ampla discussão sobre a aplicação de métodos de filtragem para frente e para trás no processamento de sinais.

Ademais foi aplicado um filtro de médias moveis para suavizar pequenas variações, facilitando a detecção de eventos importantes, como picos e complexos.

Após a aplicação do filtro, geramos os gráficos para visualizar o sinal original, o sinal filtrado e o sinal final ajustado com a remoção da média móvel feito para eliminar flutuações e suavizar o sinal, conforme a Figura 3.8. Utilizamos a biblioteca matplotlib para fazer a geração dos gráficos.

Figura 3.8 – Gráficos da amostra de sinal, primeiro sinal com ruído e os seguintes o sinal filtrados.



A remoção da componente sazonal é bem visível ao analisarmos na Figura 3.8. No gráfico, o sinal filtrado apenas pelo filtro Butterworth ainda apresenta flutuações residuais, que podem interferir na análise. Para corrigir isso, aplicamos uma média móvel que captura o comportamento sazonal do sinal e subtraímos essa média do ECG original, removendo as oscilações indesejadas.

$$ECG\_Ajustado = ECG\_Original - Media\_Movel$$

Com essa abordagem, obtemos um ECG ajustado, livre de flutuações sazonais, permitindo uma análise mais precisa das variações essenciais do sinal, como a detecção de arritmias e outras características clínicas.

3.2.4 Integração com a plataforma web e banco de dados

A integração do sistema de monitoramento, desenvolvido com o ESP8266, ocorre por meio do protocolo MQTT, com os dados coletados enviados em formato JSON Figura 3.9. Esses dados são transmitidos em tempo real e podem ser visualizados instantaneamente, proporcionando monitoramento contínuo das informações.

```
{
  "ecg": [
    308.0, 374.0, 368.0, 303.0, 360.0, 379.0, 306.0, 348.0, 384.0, 312.0,
    334.0, 388.0, 320.0, 323.0, 389.0, 334.0, 319.0, 390.0, 349.0, 312.0,
    388.0, 372.0, 314.0, 378.0, 379.0
  ]
}
```

Figura 3.9 – Modelo de Json enviado

O Node-RED desempenha um papel crucial na coleta e processamento dos dados. Ele recebe as informações do broker MQTT, processa e converte os dados antes de salvá-los no banco de dados MySQL, onde são armazenados no formato string para posterior análise.

A plataforma web, foi implementada utilizando PHP, CSS, HTML e JavaScript. Ela oferece uma interface simples com uma tela inicial de login, após a qual os usuários autenticados podem acessar gráficos gerados a partir dos dados coletados.

Além disso, a plataforma web se conecta ao broker MQTT para receber e exibir os dados em tempo real, garantindo que as informações mais recentes estejam disponíveis para visualização imediata. Simultaneamente, os usuários também podem acessar o banco de dados MySQL para visualizar dados históricos, já salvos pelo sistema, oferecendo tanto visualizações em tempo real quanto o acesso a dados previamente armazenados.

4 RESULTADOS

O estudo teve como objetivo principal desenvolver um sistema de monitoramento da saúde de animais utilizando Internet das Coisas (IoT), com foco na coleta e transmissão de dados vitais, como sinais de ECG. A pesquisa buscou não apenas a implementação técnica do sistema, mas também a eficácia na coleta e tratamento dos dados, visando aprimorar a medicina veterinária e a saúde animal.

Os resultados obtidos foram organizados em seções que abordam as conclusões e a análise dos dados coletados. Os tópicos foram divididos na análise do sinal coletado e no resultado do tratamento dos dados.

4.1 Análise do sinal coletado e tratamento dos dados

O sinal de ECG é coletado a cada 2 milissegundos, resultando em uma frequência de amostragem de 500 Hz. As amostras são armazenadas em um array temporário de 25 posições devido a limitações de memória do microcontrolador ESP8266. Quando 25 amostras são acumuladas em um tipo de texto *json*, elas são enviadas para o *broker* via protocolo MQTT.

Inicialmente foi dimensionado o envio de 500 amostras, uma vez que é um ciclo completo de coleta, com nossa frequência de amostragem a 500hz. No entanto, devido à limitação de memória do esp8266, não foi possível enviar diretamente um array de 500 posições, havendo a necessidade em outra lógica para o envio dessas amostras. No momento da coleta, as amostras são salvas e enviadas em um array temporário de 25 posições.

O sinal coletado é enviado para o broker em ciclos, com o ESP8266 atuando como o *client publish*. Nos testes realizados, foi possível enviar os dados tanto para um broker local, hospedado em uma máquina virtual (VM), quanto para um broker externo, como o *test.mosquitto*. A Figura 4.2 temos o resultado dos testes de envio em tempo real para o broker.

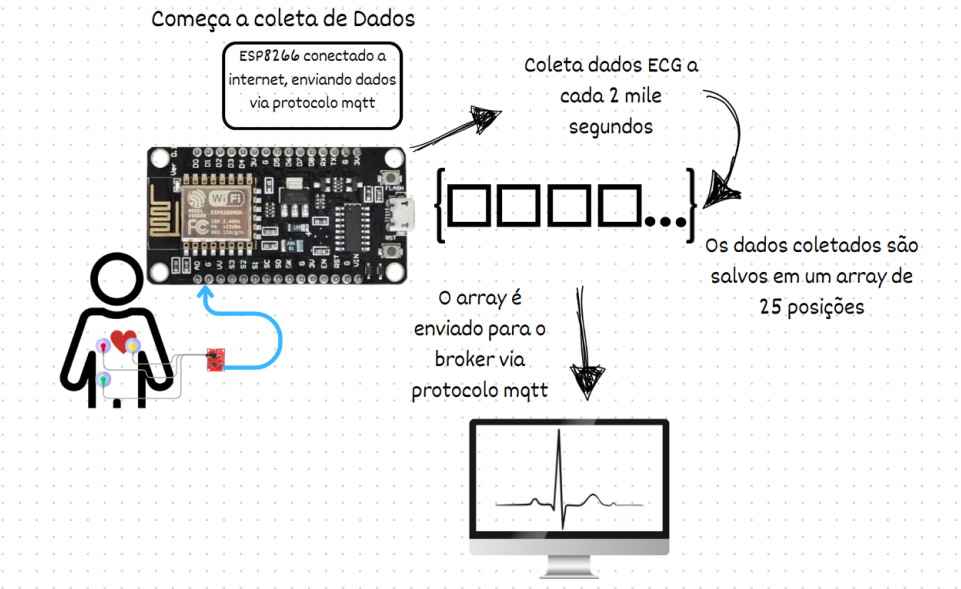


Figura 4.1 – Modelo de Coleta e envio de dados

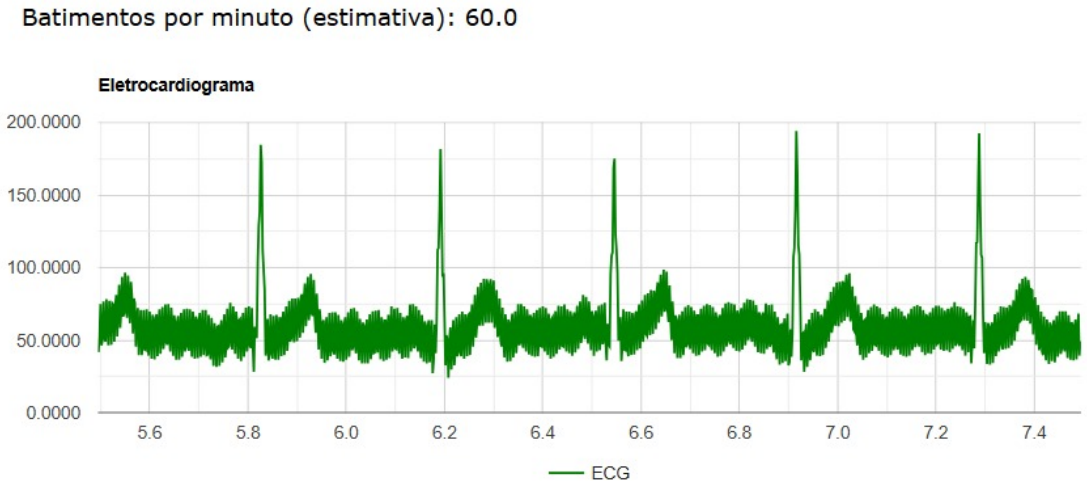


Figura 4.2 – Gráfico resultante do envio em tempo real para o broker

Em ambos os cenários, o envio foi bem-sucedido, com baixa latência e sem perda de dados. Esses resultados indicam uma comunicação estável e eficiente entre o dispositivo e o broker, independentemente da localização do servidor.

4.1.1 Tratamento dos dados

A Figura 4.3 exibe uma amostra de batimento cardíaco de um coelho, a frequência de amostragem do sinal é de 1000 Hz, onde selecionamos as primeiras 4000 amostras para testes. Utilizamos as amostras fornecidas no site da *PhysioZoo* para realizar essa etapa de testes. A imagem mostra primeiramente o sinal com ruído, marcado por variações e perturbações indesejadas. Essas perturbações são recorrentes de interferências elétricas próximas ao local ou ruído ambiental, que interferem em uma análise adequada e dificultam a realização de diagnósticos precisos.

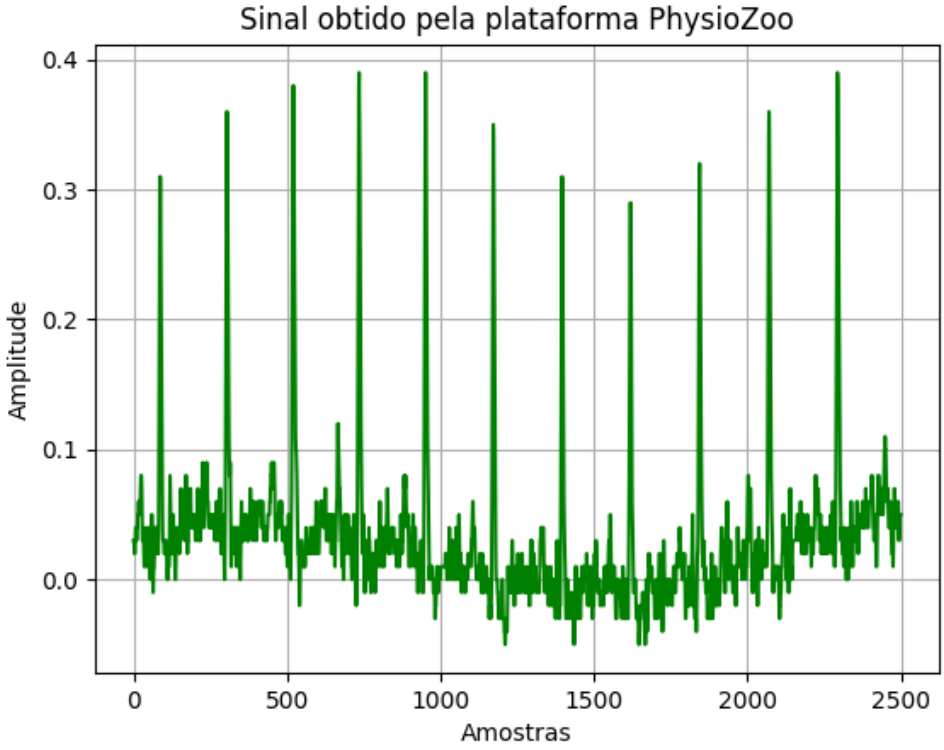


Figura 4.3 – Amostra de sinal com ruído

Bem como nos testes com o PhysioZoo o sinal coletado apresenta ruídos indesejados, que podem ser causados por interferências elétricas ou ruído ambiental Figura 4.3. Para mitigar esses efeitos, foi aplicado o filtro *Butterworth* de ordem 2, os testes obtivemos os resultados onde o filtro melhorou a clareza do sinal ao remover perturbações. Quando o sinal passa pelo filtro *Butterworth*, as frequências que estão fora da faixa de interesse são atenuadas e após o processamento, o sinal se torna mais limpo, permitindo uma análise mais precisa.

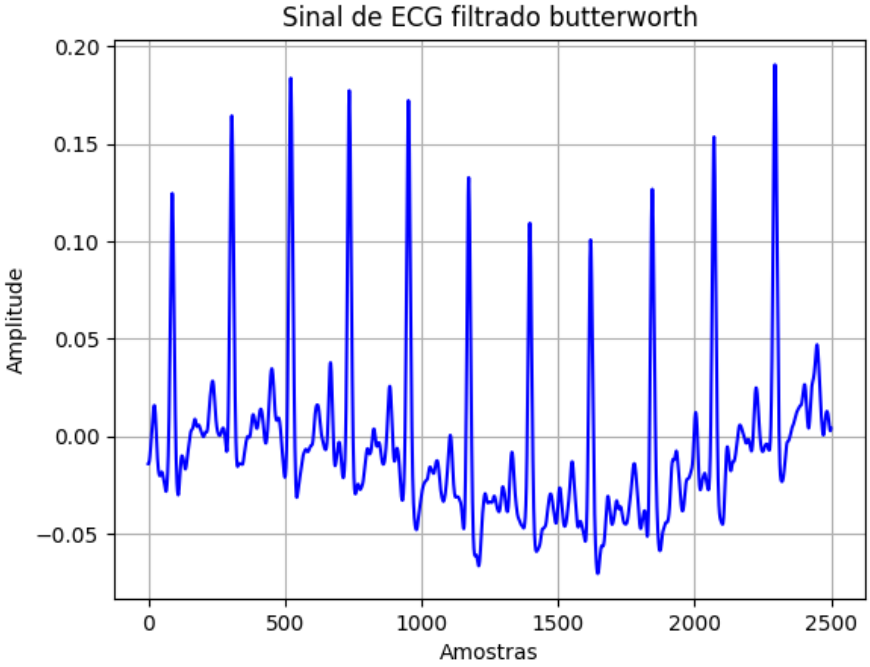


Figura 4.4 – Sinal filtrado com Butterworth

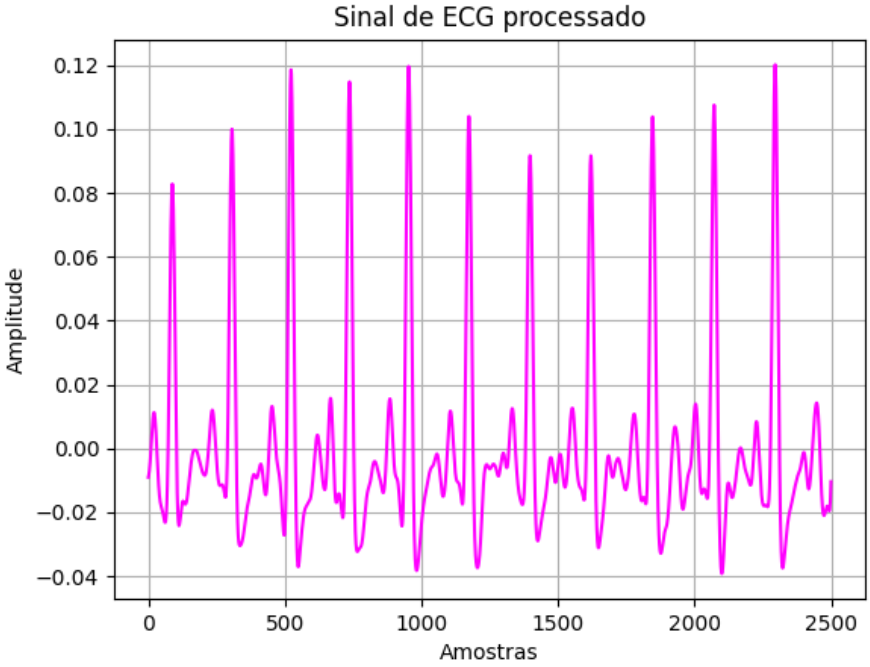


Figura 4.5 – Amostra de sinal processado

A Figura 4.4 temos o sinal de amostra obtido na plataforma PhysioZoo resultando após a aplicação do filtro, se percebe que o sinal está mais limpo, onde já é possível visualizar as ondas do círculo cardíaco, bom como o complexo QRS.

Por último temos o sinal passando pelo processo re remoção da sazonalidade, normalizando pontos necessários sem perder as características principais do sinal. O sinal processado(Figura 4.5) sai mais limpo e com menos ruído. Nesse caso, a combinação do filtro Butterworth com o filtro de médias móveis fornece um sinal suavizado e normalizado,

ideal para uma análise precisa. Agora temos um sinal mais limpo e normalizado, permitindo que especialistas façam a análise e diagnósticos mais precisos.

Os resultados obtidos a partir dos dados de amostras fornecidos pela plataforma PhysioZoo foram essenciais para o desenvolvimento deste trabalho. A realização desses testes permitiu uma análise aprofundada do desempenho do filtro e possibilitou ajustes criteriosos em seus parâmetros, assegurando que ele estivesse devidamente calibrado para atender às nossas necessidades. Com isso, conseguimos otimizar o filtro para maximizar sua eficácia e precisão, tornando-o mais adequado ao propósito deste estudo.

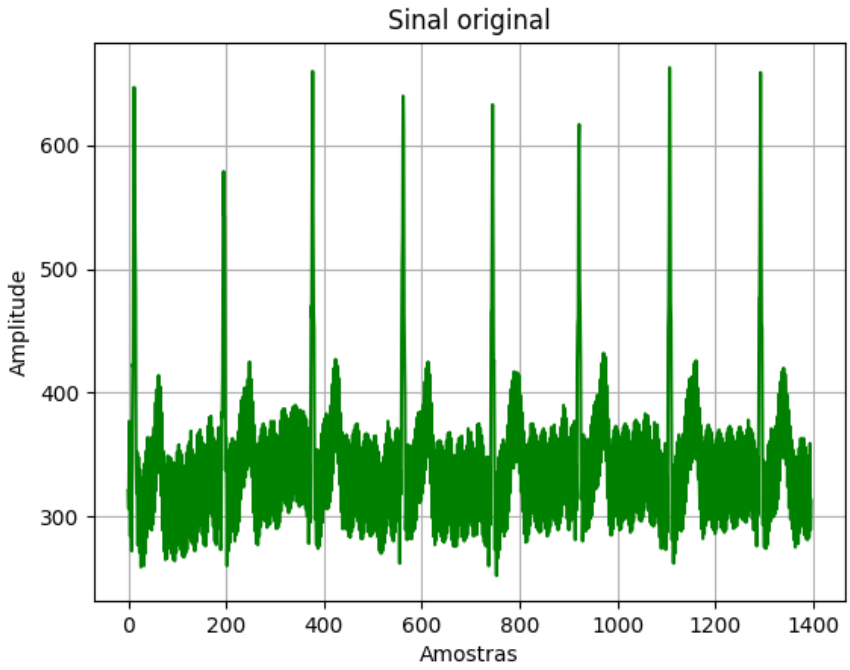


Figura 4.6 – Sinal Coletado

Após o desenvolvimento do filtro, realizamos o teste de coleta do sinal com o microcontrolador, o sinal coletado foi submetido a um processo completo de filtragem para remover flutuações irregulares e distorções tanto de amplitude quanto de frequência, aprimorando a qualidade do sinal. Na Figura 4.6, observa-se uma amostra do sinal captado pelo sensor em um indivíduo, cuja leitura foi realizada pela porta analógica do ESP, conforme descrito detalhadamente no Subseção 3.2.1. Este procedimento foi essencial para garantir a precisão e a confiabilidade dos dados utilizados ao longo deste trabalho.

Após a aplicação do filtro Butterworth de segunda ordem e do filtro de médias móveis, o sinal coletado se torna mais claro e livre de ruídos. Com forme a Figura 4.7 vemos a mesma amostra do sinal coletado e nela já temos uma clara visualização das ondas do ciclo cardíaco, comprovando a precisão e eficiência no processamento dos dados.

A Figura 4.7 apresenta a amplitude do sinal em função das amostras coletadas em um tempo T. O sinal exibido passou pelos filtros Butterworth de segunda ordem e o filtro de médias móveis. Após processado, o sinal apresenta picos que são as ondas principais do ciclo cardíaco (provavelmente os complexos QRS de um ECG). Há uma redução significativa dos ruídos em torno do sinal base, tornando o padrão mais claro e permitindo uma melhor identificação de eventos específicos.

O uso dos dois filtros conseguiu destacar os componentes relevantes do sinal cardíaco, removendo interferências. Este processo torna as análises mais precisas e confiáveis,

especialmente em aplicações médicas, como a identificação de arritmias ou anomalias cardíacas.

O uso do filtro Butterworth, associado ao filtro de médias móveis, foi indispensável para assegurar a clareza do sinal coletado. Ao eliminar as interferências de alta frequência, o sinal se torna mais estável e confiável, facilitando diagnósticos e interpretações de dados.

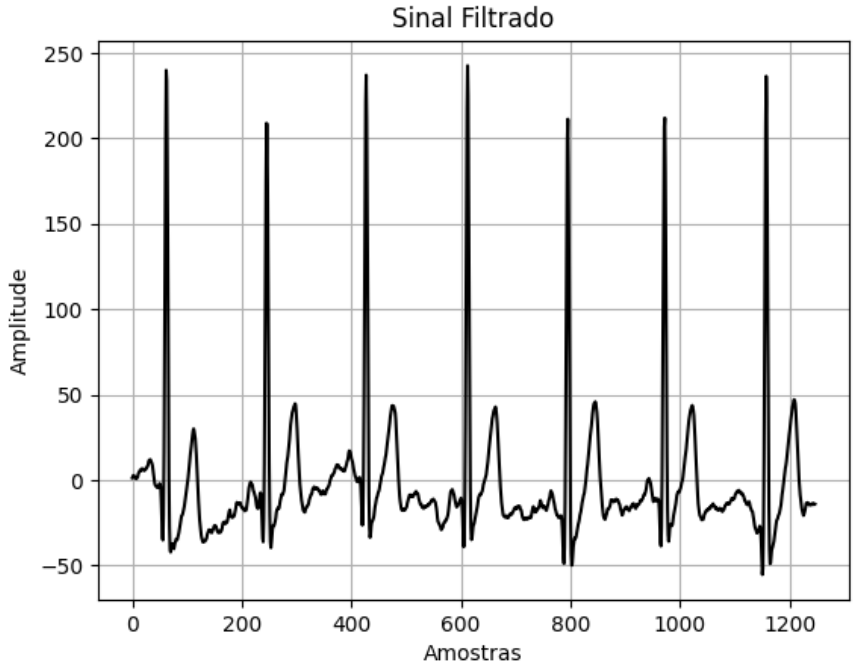


Figura 4.7 – Sinal Filtrado

A implementação do filtro com Python foi bem-sucedida, demonstrando um alto grau de precisão e eficiência no processamento dos dados. O código conseguiu aplicar as operações de filtragem com rapidez, preservando a integridade dos dados e removendo ruídos indesejados eficazmente. Como resultado, obteve-se um sinal claro e preparado para análises subsequentes ao nível profissional, evidenciando a robustez e a confiabilidade do filtro desenvolvido.

Além disso, o processo de implementação permitiu ajustes específicos que melhoraram ainda mais o desempenho do filtro, adaptando-o às características dos dados trabalhados. Essa adaptabilidade mostra que a solução não só atende aos requisitos iniciais, mas também pode ser ajustada para diversas aplicações, tornando-a uma ferramenta versátil para filtragem de dados em contextos que exigem precisão.

5 CONCLUSÃO

A pesquisa apresentada abordou o desenvolvimento de um sistema de monitoramento da saúde de animais utilizando a Internet das Coisas (IoT), aplicados sensores para coleta de dados vitais em tempo real. Esse sistema é voltado tanto para animais de pequeno quanto de grande porte, e visa auxiliar o trabalho de veterinários e pesquisadores, proporcionando uma ferramenta eficiente para o acompanhamento constante de parâmetros como temperatura corporal e frequência cardíaca. A combinação de biossensores com a tecnologia IoT oferece uma abordagem inovadora, que pode ser integrada em diversos cenários de monitoramento.

Com base nos resultados obtidos, este estudo fornece uma base para o monitoramento da saúde de animais por meio da aplicação de IoT. Embora tenham sido identificadas áreas que necessitam de ajustes, este trabalho oferece oportunidades para futuras pesquisas e aprimoramentos tanto no sistema de *hardware* quanto de *software*.

O projeto demonstrou a viabilidade de utilizar o microcontrolador ESP8266 e outros componentes de hardware e software para criar um dispositivo de baixo custo e fácil instalação. Os testes iniciais mostraram que o sistema consegue transmitir dados em tempo real para uma plataforma web, onde são tratados e armazenados para análise. Apesar das limitações encontradas no processamento dos dados devido ao baixo poder computacional do ESP8266, o dispositivo mostrou-se eficaz em fornecer uma coleta de dados satisfatória, principalmente para sinais vitais simples e monitoramento contínuo.

Vemos o trabalho explorou a importância do uso de filtros, como o filtro Butterworth e de médias móveis, para remover ruídos dos sinais de ECG, garantindo maior precisão nos dados analisados. A implementação desses filtros se mostrou crucial para a obtenção de dados mais confiáveis, mesmo em condições de coleta de dados que podem sofrer interferências. Essa etapa foi essencial para transformar os dados brutos em informações claras e utilizáveis, otimizando o valor do sistema para aplicações veterinárias e de pesquisa.

Além disso, as conclusões indicam que houve êxito na coleta e tratamento dos dados. A filtragem e remoção de ruídos no sinal de ECG demonstraram uma eficácia notável em comparação com outras tecnologias disponíveis no mercado. Esses resultados contribuem para o avanço do monitoramento da saúde dos animais, fornecendo insights valiosos para aprimoramentos e aplicações futuras neste campo.

Em suma, o trabalho fornece uma base sólida para o desenvolvimento de tecnologias IoT aplicadas ao cuidado animal, mostrando que o monitoramento remoto e em tempo real de dados vitais é uma possibilidade prática e benéfica. A continuidade desse projeto tem potencial para ampliar significativamente a aplicação da IoT na saúde animal, promovendo avanços no bem-estar dos animais e oferecendo novas ferramentas para o campo da medicina veterinária e conservação.

5.1 Trabalhos Futuros

- Expansão para Monitoramento Multissensorial; Desenvolver uma versão do sistema que integre outros sensores biométricos, como oxímetro, temperatura e pressão arterial, para monitoramento de múltiplos sinais vitais em tempo real. Esse projeto seria útil para criar uma plataforma mais completa para monitoramento remoto de saúde, podendo ser aplicada tanto em animais quanto em humanos.
- Integração com Sistemas de Prontuário Eletrônico; Implementar uma integração com sistemas de prontuário eletrônico (EHR) para armazenar e sincronizar os dados

de monitoramento com o histórico médico completo dos pacientes. Esse projeto facilitaria o acompanhamento longitudinal e o compartilhamento de informações relevantes com profissionais de saúde.

- Implementação de Algoritmos de Análise de Dados em Tempo Real; Integrar algoritmos de Machine Learning ou inteligência artificial para análise em tempo real dos dados de ECG e detecção de anomalias. Por exemplo, identificar arritmias ou padrões atípicos que possam servir como alerta para intervenção precoce.

REFERÊNCIAS

ASHTON, K. That 'internet of things' thing: In the real world, things matter more than ideas. **RFID Journal**, RFID Journal, n. 1, 1999. Disponível em: <https://www.rfidjournal.com/that-internet-of-things-thing>. Citado na página 12.

COSTA, W. G. A.; COSTA, C. de M.; REGIS, C. D. M. Detecção do complexo qrs utilizando o algoritmo de pan & tompkins modificado. In: SBRT. **Anais do XXXIII Simpósio Brasileiro de Telecomunicações - SBrT2015**. Juiz de Fora, MG, 2015. Citado na página 14.

CUNHA, P. **Um Modelo de Eletrocardiógrafo Portátil de Baixo Consumo**. Dissertação (Dissertação de Mestrado) — Universidade Federal de Alagoas, Maceió, 2012. Instituto de Computação, Modelagem Computacional de Conhecimento. Citado na página 17.

GUSTAFFSON, F. Determining the initial states in forward-backward filtering. **Transactions on Signal Processing**, v. 46, p. 988–992, 1996. Citado na página 29.

HIVEMQ. **MQTT Essentials - Part 1: Introducing MQTT**. 2024. Acessado em: 14 de novembro de 2024. Disponível em: <https://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt/>. Citado na página 19.

KARTHICK, G.; SRIDHAR, M.; PANKAJAVALLI, P. Internet of things in animal healthcare (iotah): Review of recent advancements in architecture, sensing technologies and real-time monitoring. **SN Computer Science**, Springer Nature, v. 1, n. 301, 2020. Disponível em: <https://doi.org/10.1007/s42979-020-00310-z>. Citado na página 12.

NODE-RED. **User Guide - Concepts**. 2024. Acessado em: 14 de novembro de 2024. Disponível em: <https://nodered.org/docs/user-guide/concepts>. Citado na página 19.

QUADROS, A.; SILVA, J.; SANDOVAL, ; LUZ, R.; NEVES, T.; SILVA, V.; COSTA, G.; PAREDES, J.; UZELAC, I.; SALINET, J. **Estimação de sinais cardíacos em um modelo animal pela técnica iECG**. São Bernardo do Campo–SP, Brasil: [s.n.]. S.D. Citado na página 15.

ROCHA, M. **Desenvolvimento Open-Source para a Internet das Coisas (Arquiteturas para Interfaces Web e Móvel)**. Niterói: [s.n.], 2018. Citado na página 18.

SciPy Developers. **scipy.signal.butter - SciPy v1.10.0 Manual**. 2024. Acessado em: 14 de novembro de 2024. Disponível em: <https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.butter.html#scipy.signal.butter>. Citado na página 28.

SciPy Developers. **scipy.signal.filtfilt - SciPy v1.10.0 Manual**. 2024. Acessado em: 14 de novembro de 2024. Disponível em: <https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.filtfilt.html#scipy.signal.filtfilt>. Citado na página 28.

SHEMLA, O.; BEHAR, J. Physiozoo-mammalian nsr databases (version 1.0.0). **PhysioNet**, PhysioNet, 2019. Disponível em: <https://doi.org/10.13026/p63q-hq95>. Citado na página 22.

Universidade Estadual Paulista (UNESP). **Optoeletrônica - Capítulo 3: Seção 3.4**. 2024. Acessado em: 14 de novembro de 2024. Disponível em: <https://www.feis.unesp.br/Home/departamentos/engenhariaeletrica/optoeletronica/capitulo-3---secao-3.4-loe.pdf>. Citado na página 16.