UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

DEPARTAMENTO DE COMPUTAÇÃO

Sistema de Aquisição, Armazenamento e Transmissão em Tempo Real de Sinais Musculares

Trabalho de Conclusão de Curso

Luiz Henrique Leite Paes da Costa

São Cristóvão – Sergipe

2021

UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

DEPARTAMENTO DE COMPUTAÇÃO

Luiz Henrique Leite Paes da Costa

Sistema de Aquisição, Armazenamento e Transmissão em Tempo Real de Sinais Musculares

Trabalho de Conclusão de Curso submetido ao Departamento de Computação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de Bacharel em Engenharia de Computação.

Orientador(a): Edward David Moreno Ordonez

São Cristóvão – Sergipe

2021

Agradecimentos

Inicio os meus agradecimentos em reconhecimento a todo incentivo, apoio, amor e companheirismo que minha mãe, Vânia, tem compartilhado comigo durante toda a minha jornada. Se estou onde estou, é por causa de ti. Ao meu irmão Leandro, sou grato pela amizade e companheirismo de sempre. Agradeço também ao meu pai Luiz e aos meus irmãos Giovanny e Lorenzo por estarem do meu lado sempre, mesmo à distância.

Agradeço a minha namorada, amiga e companheira de todos os momentos, Brenda, pelo incentivo neste projeto e em todos os outros da minha vida.

Obrigado aos meus avós maternos, Pedro (in memoriam) e Jovelina, e aos meus avós paternos, Veríssimo (in memoriam) e Dona Sinhá (in memoriam), por sempre terem sido sinônimo de acolhimento e cuidado ao decorrer da minha vida. E se tratando de família, estendo carinhosamente os meus agradecimentos aos meus padrinhos, Vivi e Gilson, e aos meus primos pela amizade e infância que vivemos juntos.

Aos meus amigos Paulo Victor e Yan gostaria de agradecer pelos momentos que compartilhamos e pela parceria de longa data que cultivamos até hoje. Um obrigado, igualmente especial, a William, meu amigo fiel desde a infância. Agradeço também a todos aqueles amigos que fiz ao decorrer da graduação, em especial a Michel, Reneilson e José Lucas, pelas ideias, experiências e aprendizados que dividimos e amadurecemos juntos. Sem vocês essa caminhada não seria a mesma!

Ao meu orientador, o prof. Edward, agradeço pelo apoio e oportunidades que me concedeu e que me permitiram finalizar mais essa etapa.

Por fim, reitero aqui a minha gratidão a todos aqueles que acreditaram em mim e fizeram parte da minha história.

Resumo

Com o advento cada vez maior de pessoas à prática de atividades físicas e baixa oferta de profissionais de saúde, se torna cada vez mais difícil o acompanhamento individual e presencial de pacientes e praticantes de atividades físicas. Uma maneira de observar a eficiência na execução de exercícios físicos é através da captação de sinais elétricos gerados pelos músculos por meio de eletromiografia, avaliando tais sinais em tempo real ou o histórico de um paciente para ter ciência da evolução do mesmo. O presente trabalho tem como objetivo desenvolver um sistema de captação e armazenamento permanente de sinais musculares de um paciente para a visualização dos mesmos em tempo real ou do histórico por parte dos profissionais de saúde, visando a identificação de problemas no desenvolvimento dos músculos do paciente ou até mesmo possíveis doenças.

Palavras-chave: eletromiografia; armazenamento de sinais musculares; sistema de captação de sinais musculares.

Abstract

With the increasing number of people practicing physical activities and a low supply of health professionals, it is becoming increasingly difficult to monitor individual and face-to-face patients and practitioners of physical activities. One way to observe the efficiency in the execution of physical exercises is by capturing electrical signals generated by the muscles through electromyography and evaluating them in real-time or a patient's history, to be aware of the evolution of the same. The present work aims to develop a system for the capture and permanent storage of a patient's muscle signals for the visualization of them in real-time or history by health professionals, aiming at the identification of problems in the development of the patient's muscles or even possible diseases.

Keywords: electromyography, storage of muscle signals, system for capturing muscle signals.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ACID	atomicity, consistency, isolation, durability
AOT	Ahead of Time
API	Application Programming Interface
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DNS	Domain Name Server
DOM	Document Object Model
DTW	Dynamic Time Wraping
ECG	Eletrocardiografia
EMG	Eletromiografia
FPGA	Institute of Electrical and Electronics Engineers
GPS	Global Positioning System
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
JSON	JavaScript Object Notation
LCD	Liquid-crystal Display
MMG	Mecanomiografia
MQTT	Message Queuing Telemetry Transport
PC	Personal Computer
PPG	Photoplethysmography
RAM	Random Access Memory

REST	Representional State Transfer
SDK	Software Development Kit
SD	Secure Digital
SGBD	Sistema Gerenciador de Banco de Dados
SoC	System on Chip
SPI	Serial Peripheral Interface
SQL	Structured Query Language
SVM	Support Vector Machine
TCP	Transmission Control Protocol
UI	User Interface
URL	Uniform Resource Locator
USB	Universal Serial Bus
XML	eXtensible Markup Language

Sumário

1 Introdução

1.1 Apresentação

A prática de atividades físicas tem crescido exponencialmente, tornando-se um fator eminente na sociedade ao passo que é alvo de tantas atenções. Seja por motivos de saúde ou de estética, este hábito tem expandido um mercado já promissor que, anteriormente, se concentrava em um nicho de consumidores específicos cujos objetivos são ainda mais específicos, como é o caso de atletas e competidores profissionais de fisiculturismo (EDINGTON et al., 2016). Porém, diante da procura de um público muito mais amplo e diversificado, e cada vez mais engajado na busca por saúde e estética, surge a necessidade de educar e entender o desenvolvimento motor dos indivíduos durante a prática dos exercícios físicos, a fim de evitar problemas com lesões à medida que também torna-se possível reduzir os riscos relacionados a uma série de doenças, principalmente àquelas de origem cardiovascular (TROST; BLAIR; KHAN, 2014).

Seja para praticantes principiantes ou avançados, há uma necessidade de monitorar os movimentos executados a fim de obter maior êxito perante os objetivos designados. Nesse cenário, a demanda pelos serviços prestados por profissionais de educação física cresce de modo desproporcional à disponibilidade ofertada pelos empregadores, dificultando um acompanhamento preciso e adequado para cada praticante de exercícios (PIND; MäESTU, 2018).

Além da aplicabilidade na prática de atividades físicas, o monitoramento de sinais fisiológicos é bastante utilizado na fisioterapia, uma vez que esta ciência é de suma importância no tratamento de disfunções cinéticas funcionais tanto de órgãos quanto de sistemas (SARAEE et al., 2017). Para o caso específico no qual as lesões ocorreram ou não em decorrência de acidentes relacionados à prática de atividades físicas, a fisioterapia é um dos meios mais apropriados no treino de recuperação muscular. No entanto, como se trata de situações mais delicadas, uma vez que a lesão já existe, os serviços prestados por profissionais qualificados são considerados essenciais para o desenvolvimento das atividades de recuperação, em paralelo com o requerimento

de aparelhos mais sofisticados de monitoramento, o que causa um aumento direto no custo de tratamentos fisioterapêuticos (RIGHETTI et al., 2020).

Dados os problemas de custo e acesso existentes nas duas situações expostas até então, um dos modelos de propostas que vem ganhando espaço é a utilização de dispositivos *wearables* ("vestíveis", numa tradução livre) que, por serem constituídos por sensores, monitoram diversos parâmetros fisiológicos do corpo humano durante a prática das mais diversas atividades musculares e, ainda, possibilitam o fornecimento de determinadas informações ou instruções específicas baseadas no processamento dos dados adquiridos (WETMORE, 2016).

Nesse contexto, e para ambas as atividades citadas, um tipo de órgão constantemente monitorado é o músculo do tipo esquelético, que além de ser extremamente exigido em atividades físicas, é o foco de recuperação da lesão em muitos casos de tratamentos fisioterapêuticos. Por consequência, existem diversas soluções para o acompanhamento dessas atividades, seja de forma mais complexa, com o emprego de aparelhos mais sofisticados e de alto custo; ou na forma de dispositivos *wearables*, que possuem, de modo geral, uma proposta inicial baseada no barateamento de despesas, na praticidade e na personalização, mas que, apesar do seu grande potencial, ainda se mostram com muitas lacunas a serem preenchidas (TEDESCO et al., 2019).

1.2 Motivação

A motivação do presente trabalho consiste na já citada necessidade de monitoramento automatizado de sinais vitais durante a prática de atividades físicas, tanto por praticantes iniciantes, quanto por atletas de alto rendimento, partindo da perspectiva das necessidades apresentadas por cada um desses dois grupos. Enquanto a necessidade do primeiro grupo, referente aos iniciantes, diz respeito, sobretudo, à execução correta ou não dos movimentos, apesar de, ainda assim, não dispensar a presença de um profissional da educação física devidamente capacitado; as vantagens propostas para o segundo grupo, composto por atletas, irão além, de modo a qualificar o desempenho alcançado e servindo como um parâmetro para estabelecer as devidas alterações no treinamento que otimizarão os resultados pretendidos.

Uma outra motivação concerne em atribuir mais praticidade e menor custeamento na verificação dos movimentos realizados por pacientes que se encontram em trabalho de recuperação muscular através da fisioterapia. Atualmente, essa verificação é feita, comumente, pelo intermédio de aparelhos de alto custo e complexidade, ou, até mesmo, através de um monitoramento empírico, como, por exemplo, uma simples observação acerca da capacidade motora do paciente.

1.3 Objetivos

1.3.1 Objetivo Geral

Objetivo deste trabalho é desenvolver um sistema de captação, armazenamento permanente e exibição via interface gráfica de sinais musculares (EMG) para que, desse modo, possam ser realizadas avaliações clínicas à distância de um paciente para alcançar determinados objetivos, como avaliação de hipertrofia muscular, acompanhamento de recuperação de lesões via fisioterapia e identificação de doenças.

1.3.2 Objetivos Específicos

- Desenvolver dispositivo que detecte o sinal muscular de um membro do corpo humano e transmita essa informação para o servidor de armazenamento;
- Desenvolver software que armazene amostras de um sinal muscular e informações do paciente, assim como disponibilize essas informações para outros softwares poderem consumi-las;
- Desenvolver aplicativo web que exponha graficamente a atividade muscular de determinado usuário numa faixa de tempo selecionada através interface gráfica do mesmo;
- Testar resultados desse sistema como um todo e compara-los com resultados encontradas na literatura.

1.4 Metodologia e Cronograma de Atividades

Para alcançar os objetivos pretendidos, o presente trabalho foi iniciado com o estudo da utilização dos possíveis sensores e outros dispositivos relacionados para compor o sistema embarcado de captação dos sinais desejados: movimentação corporal e utilização do músculo do paciente.

Em seguida, foi avaliada a melhor alternativa de protocolo de rede em termos de eficiência e custo de transmissão para o envio dos sinais captados pelos sensores para o software responsável pelo armazenamento permanente dos sinais musculares. Logo após, foram avaliados os tipos de banco de dados mais apropriados para o armazenamento das amostras de sinal e dos dados do paciente, assim como a melhor maneira de disponibilizar os dados para posteriores aplicações.

Após isso, foi implementado o software de armazenamento dos sinais, estabelecendo sua comunicação com os bancos de dados previamente selecionados e a interface que o mesmo disponibilizará para as aplicações que consumirão os dados.

Por último, foi iniciada a implementação da aplicação web que consome as informações relacionadas ao paciente e seu histórico de sinal muscular, através da utilização do software de armazenamento.

1.5 Organização do Trabalho

Para um melhor entendimento do trabalho realizado, esta obra foi dividida nos seguintes capítulos:

- Capítulo 1 - Introdução: aborda as informações básicas necessárias para o entendimento do trabalho, assim como as motivações para a realização do mesmo e sua relevância perante trabalhos existentes;
- Capítulo 2 - Fundamentação Teórica: expõe os conceitos teóricos e fundamentos para o entendimento da execução do trabalho;
- Capítulo 3 - Trabalhos Relacionados: analisa a literatura e produtos relacionados ao trabalho proposto, visando ter um alicerce sólido para o desenvolvimento da proposta, assim como para evitar a repetição de experimentos já bem sucedidos e contribuir com o estado da arte;
- Capítulo 4 - Proposta e Implementação do Sistema MyoVita : são detalhados os procedimentos necessários para o desenvolvimento do trabalho e avaliados se os objetivos estabelecidos foram alcançados;
- Capítulo 5 - Conclusão: compara os resultados obtidos na implementação com os encontrados na literatura, a fim de verificar a eficácia da automação nesse processo de acompanhamento de atividade física. Além disso, serão feitas propostas de continuidade e melhorias deste trabalho.

2

Fundamentação Teórica

2.1 Bancos de Dados Relacionais e de Série Temporal

De acordo com Silberschatz, Korth e Sudarshan (2020), um banco de dados é uma coleção organizada de dados armazenados e acessados eletronicamente. A maioria dos bancos de dados são componentes de um sistema maior, que é o SGBD (Sistema de Gerenciamento de Banco de Dados). Um SGBD engloba o banco de dados em si e interfaces que permitam a utilização do mesmo por outros sistemas de software ou usuários finais (ELMASRI; NAVATHE, 2016).

Os SGBDs podem ser classificados de acordo com os modelos de banco de dados que suportam, sendo classificados em relacionais e não-relacionais. Os bancos de dados relacionais tornaram-se extremamente populares a partir da década de 1980 e, por isso, há essa distinção explícita entre esse tipo de modelagem e as demais. Os bancos relacionais, em sua grande maioria, usam SQL (Structured Query Language) como linguagem para escrever e consultar dados, o que levou os bancos de dados não relacionais a ficaram conhecidos como NoSQL, já que usam diferentes linguagens de consulta (GARCIA-MOLINA; ULLMAN; WIDOM, 2009).

Os bancos relacionais têm seus dados modelados como linhas e colunas em uma série de tabelas, de tal forma que uma tabela representa uma entidade que se deseja modelar, a exemplo de um sistema de gerenciamento de clientes, no qual a entidade cliente seria uma tabela. Todos os clientes para o escopo daquele sistema provavelmente possuem propriedades em comum como nome, idade e outros dados. A definição de quais podem ser as propriedades de uma tabela são definidas pelas colunas da mesma. Sendo assim, no exemplo dado, nome e idade são colunas (ou campos) da tabela e cada cliente registrado no banco é chamado de "registro"ou "linha"da tabela cliente. Cada tabela no banco relacional possui quase sempre uma coluna cujo objetivo é identificar unicamente um registro, chamado de chave primária. Além disso, entidades diferentes podem se relacionar neste tipo de banco de dados através de chaves estrangeiras, que nada mais

são do que um tipo de coluna cujo dado aponta para outra tabela. No sistema dado como exemplo até então, uma tabela denominada "compra"possuiria uma coluna chamada "id_do_cliente"que a conectaria com o registro da tabela "cliente"com a qual ela se relaciona. A Figura 1 ilustra a situação de estruturação das tabelas utilizada como exemplo.

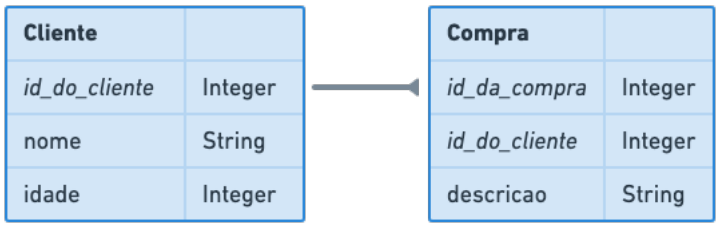

Figura 1 – Exemplo de tabelas relacionadas em um banco relacional

Uma das principais vantagens dos SGBDs relacionais é que eles provêm transações com características ACID: atomicidade, que garante que cada transação seja tratada como uma única "unidade", a qual pode ser bem-sucedida ou falhar completamente; consistência, que garante que uma transação só pode trazer o banco de dados de um estado válido para outro; isolamento, que assegura que a execução simultânea de transações deixe o banco de dados no mesmo estado que teria sido obtido se as transações fossem executadas sequencialmente; e durabilidade, que se certifica que uma vez que uma transação foi confirmada, a mesma permanecerá confirmada até em caso de falha do sistema (SILBERSCHATZ; KORTH; SUDARSHAN, 2020).

Por outro lado, de acordo com Sadalage e Fowler (2013), os bancos não-relacionais englobam todos os tipos de banco que não usam uma modelagem de dados via tabelas. As motivações para esta abordagem incluem simplicidade de design, maior compatibilidade de modelagem com linguagens de programação orientadas a objeto (para facilitar a conversão de um objeto para o formato de entidade adotado pelo banco de dados), assim como maior facilidade que os bancos relacionais para fazer escalonamento horizontal (que consiste em criar mais de uma instância de execução SGBD para suportar maior volume de carga).

As estruturas de dados usadas por bancos de dados NoSQL (por exemplo, par de valores-chave, coluna larga, gráfico ou documento) são diferentes daquelas usadas por padrão em bancos de dados relacionais, tornando algumas operações mais rápidas em NoSQL. Algumas vezes, as estruturas de dados utilizadas pelos bancos de dados NoSQL também são vistas como "mais flexíveis"do que as tabelas de banco de dados relacionais, sendo que a maioria dos armazenamentos NoSQL não possui transações ACID verdadeiras e as características específicas de um determinado banco de dados NoSQL depende do problema que ele se propõe a resolver (TIWARI, 2011).

Um tipo de banco de dados não-relacional é o de série temporal (time series databases), que são otimizados para armazenar pares de tempo e valor. Esse tipo de banco normalmente separa o conjunto de características fixas e discretas de seus valores dinâmicos e contínuos em

conjuntos de pontos ou “tags”. Um bom exemplo disso é o armazenamento de utilização da CPU para monitoramento de desempenho: as características fixas incluiriam o nome “Utilização da CPU”, as unidades de medida em porcentagem (%), e um intervalo de 0 a 1, e os valores dinâmicos armazenariam a porcentagem de utilização e data/hora (DUNNING et al., 2014).

Um exemplo de banco de dados de série temporal é o InfluxDB, o qual não possui dependências externas e fornece uma linguagem semelhante a SQL para efetuar suas transações, com funções centradas no tempo integradas para consultar uma estrutura de dados composta de medidas, séries e pontos. Cada ponto consiste em vários pares de chave-valor, denominados de fieldset e timestamp que, quando agrupados por um conjunto de pares de valores-chave, os chamados conjunto de tags, definem uma série. Finalmente, as séries são agrupadas por um identificador de string para formar uma medida (DIX, 2020).

Ainda de acordo com Dix (2020), os valores do InfluxDB podem ser inteiros de 64 bits, pontos flutuantes de 64 bits, strings e booleanos. Os pontos são indexados por seu tempo e tagset. As políticas de retenção são definidas em uma medição e controlam como os dados são reduzidos e excluídos. Consultas contínuas são executadas periodicamente, armazenando resultados em uma medição de destino.

2.2 Protocolos de rede: MQTT, HTTP e WebSocket

Em softwares distribuídos, ou seja, que possuem componentes que se comunicam através de uma rede de computadores, há diversos padrões de comunicação. Dentro do modelo TCP/IP existem diversas opções de protocolo a serem aplicados na camada de aplicação e cada um possui características que favorecem seus usos em determinados cenários (PUDER; ROMER; PILHOFER, 2011).

O protocolo mais conhecido e utilizado, graças à Word Wide Web, é o protocolo HTTP (Hypertext Transfer Protocol), utilizado numa arquitetura cliente-servidor, na qual um software servidor possui os recursos desejados e aguarda por requisições de softwares clientes por esses recursos que, por sua vez, podem ser arquivos e objetos em diversos formatos (JSON, XML, etc.), sendo enviados quando é aberta uma sessão entre um cliente e o servidor. Uma sessão HTTP é iniciada quando um cliente envia uma requisição de um recurso ao servidor, enviando um conjunto de informações que identificam o recurso através de uma URL (Uniform Resource Locators) e as permissões que esse cliente tem sobre o recurso, entre outros. Uma vez que a transação é finalizada, sendo um caso de erro ou sucesso, a comunicação, e portanto a sessão, entre cliente e servidor é encerrada (GOURLEY; TOTTY, 2002).

As requisições HTTP podem ter diversos objetivos, a saber: busca de recurso, envio de recurso do cliente ao servidor, atualização de recurso já existente no servidor e remoção de recurso. Para tanto, existem diversos métodos de requisição, dos quais os mais utilizados são GET (para busca de recurso), POST (para envio de recurso), PUT (para atualização de recurso) e

DELETE (para remoção de recurso). Para exemplificar de modo simplificado uma requisição para busca de um recurso existente, tem-se: GET http://dominio-do-servidor.com/resource. Na Figura 2 é possível visualizar uma simplificação do fluxo de uma requisição HTTP (GOURLEY; TOTTY, 2002).

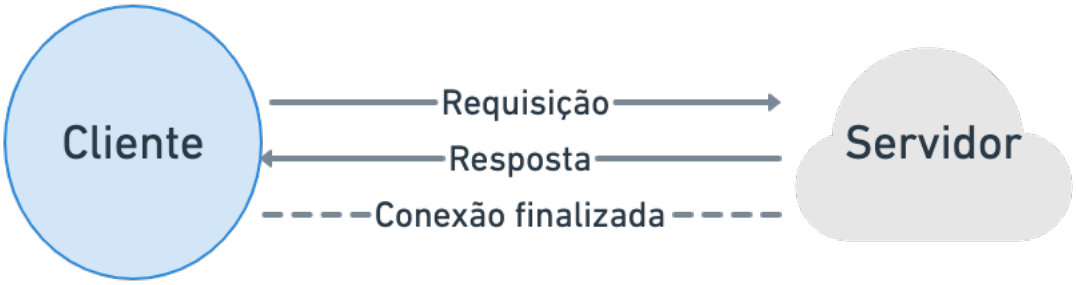

Figura 2 – Fluxo de uma requisição do protocolo HTTP

O protocolo WebSocket, ao contrário do HTTP, proporciona uma comunicação constante entre dois softwares diferentes. Uma vez que é aberta uma conexão entre dois softwares (esse processo é feito via requisição HTTP), o canal de comunicação fica aberto para as duas aplicações trocarem fluxo de dados (mensagens) até que uma das duas encerre a conexão ou fique indisponível. Essa lógica, diferente do HTTP, permite que atualizações constantes sejam transmitidas de um software a outro sem que haja a necessidade do que se chama de polling, que são requisições consecutivas com o propósito de saber se há alguma atualização no recurso desejado. A Figura 3 traz a representação de um fluxo simplificado de uma conexão com WebSocket (LOMBARDI, 2015).

Figura 3 – Fluxo de uma requisição do protocolo WebSocket

Por outro lado, o MQTT (Message Queuing Telemetry Transport), de acordo com Hillar (2017), é um protocolo de camada de aplicação que tem como objetivo ser mais leve que o HTTP e outros protocolos disponíveis para a camada de aplicação. Apesar de geralmente ser usado sobre o protocolo TCP, o MQTT consegue ser executado sobre qualquer protocolo de camada de transporte que forneça uma conexão bidirecional, ordenada e sem perdas.

Apesar de possuir "queuing"(fila) em seu nome, é apresentado em Pulver (2019) que o MQTT não utiliza filas para sua operação. Ele funciona baseado no padrão Publish-Subscribe, onde a troca de informação entre uma entidade de software e outra se dá a partir de três componentes: uma entidade publicadora (publisher), que envia determinada informação com uma forma de descrição chamada "tópico"; um broker, que verifica as requisições enviadas pelo publicador e as envia para as entidades "subscribers"(inscritos), que estão escutando aquele tópico e aguardando para respondê-los de acordo. Se não houver entidades subscribers, o broker simplesmente descarta as mensagens na sua operação padrão. Em comparação com o protocolo HTTP, percebe-se que a execução do envio de uma requisição é mais simples e menos burocrática, tanto para o publisher quanto para o subscriber, já que não envolve os tempos de verificação e abertura de comunicação que uma requisição HTTP exige. Por exigir menos recursos e tempo de processamento, esse protocolo é bastante utilizado na comunicação de sistemas embarcados com outros softwares, assim como em várias aplicações de IoT (Internet of Things). Um fluxo mais detalhado de requisições MQTT pode ser verificado na Figura 4.

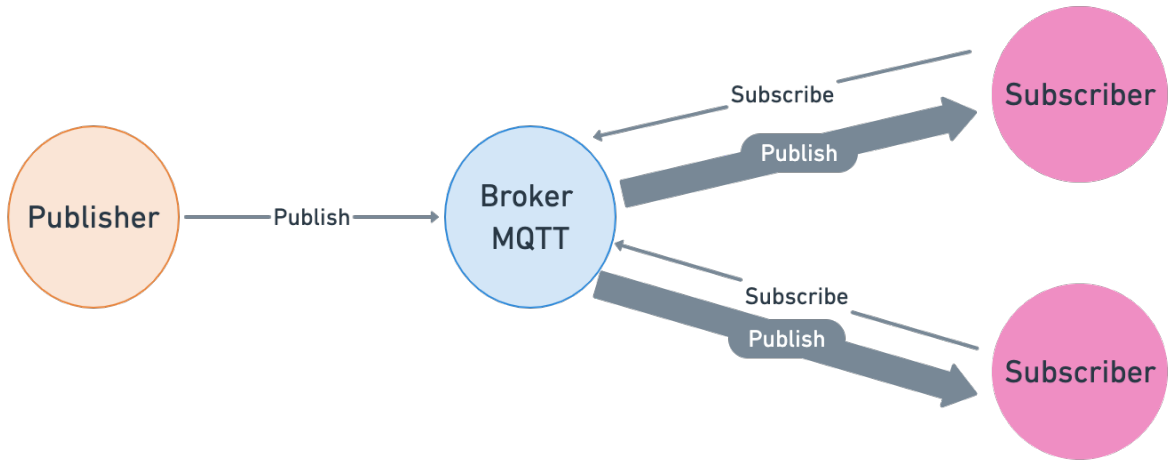

Figura 4 – Fluxo de uma requisição do protocolo MQTT

2.3 REST

Representational State Transfer (REST) é um estilo de arquitetura de software cujo objetivo é definir uma forma robusta e organizada para sistemas ditribuídos de hipermidia (como softwares da Web) se comunicarem. Esse estilo determina um conjunto de formatos de comunicação que enfatizem características como escalabilidade e baixo acoplamento da implementação dos componentes, interfaces uniformes entre os componentes, entre outras. Alguns princípios que podem ser seguidos para alcançar essas características são o uso de uma arquitetura cliente-servidor como modelo de padrão arquitetural, a ausência de estado no software servidor, a possibilidade de armazenamento em cache das requisições e recursos trocados, uso de um sistema em camadas e suporte para código sob demanda (MASSé; MASSé, 2012).

De acordo com Richardson e Amundsen (2013), softwares que seguem as restrições da arquitetura REST são conhecidos como RESTful. Como REST é voltada para aplicações na Web, um serviço RESTful deve fornecer recursos comuns na Web (como documentos, imagens, etc.) através de uma representação textual e permitir que eles sejam lidos e modificados com um protocolo sem estado e um conjunto predefinido de operações. Por exemplo, uma API RESTful baseada em HTTP (que é um protocolo sem estado) é definida pelos seguintes componentes: i. uma URI base (Ex.: http://api.example.com/); ii. métodos/operações HTTP padrão (GET, POST, PUT e DELETE); e iii. uma definição do tipo de mídia (Ex.: atom, microformats, application/vnd.collection+json, etc.) (MASSé; MASSé, 2012).

2.4 JavaScript e Node.js

JavaScript é uma linguagem de programação de alto nível, multiparadigma e geralmente compilada just-in-time. Possui sintaxe de chaves, tipagem dinâmica, orientação a objetos baseada em protótipo e funções de primeira classe. Essa linguagem possui APIs para trabalhar com textos, datas, expressões regulares, diversas estruturas de dados e o DOM (Document Object Model) para a interação com uma página HTML (HAVERBEKE, 2015).

Sua implementação e versões seguem a especificação ECMAScript. O padrão ECMAScript define apenas a sintaxe da linguagem e os recursos relacionados à mesma. As APIs e bibliotecas para uso de recursos externos, como conexão por rede, sistema de arquivos, etc. são disponibilizadas pelo ambiente de execução da linguagem, seja um navegador web ou outro sistema de tempo de execução (CROCKFORD, 2008).

Junto com HTML e CSS, JavaScript é uma das principais tecnologias da World Wide Web e foi originalmente projetada e usada apenas em navegadores da web. Cada navegador possui um conjunto de componentes de software, como um renderizador de páginas HTML e o motor de interpretação e execução de JavaScript. O navegador Google Chrome da empresa Google tem o V8 como seu motor de execução de JavaScript. Pelo V8 ser um software de código aberto, em 2009 Ryan Dahl começou o projeto hoje conhecido como Node.js, que tem como objetivo fazer o V8 er utilizável num contexto de execução que não o de um navegador (HAVERBEKE, 2015).

Até a data deste trabalho, o projeto do Node.js engloba diversos componentes, inclusive um motor de execução para a linguagem JavaScript baseado no V8, assim como um conjunto de bibliotecas e funções com diversas funcionalidades, como acesso ao sistema de arquivos, rede (via DNS, HTTP e TCP), criptografia e um gerenciador padrão de pacotes denominado npm. O npm, por sua vez, é constituído por um cliente de linha de comando, de mesma denominação, e um banco de dados online de pacotes públicos e privados pagos, conhecido como registro npm. O registro é acessado por meio do cliente e os pacotes disponíveis podem ser navegados e pesquisados no site do npm. O gerenciador de pacotes e o registro são gerenciados pela npm, Inc

(CANTELON et al., 2014).

O Node.js possui uma arquitetura orientada a eventos capaz de prover entradas e saídas assíncronas. Essas opções de projeto do ambiente visam otimizar o rendimento e a escalabilidade em aplicativos da web com muitas operações de entrada e saída, bem como para aplicativos da Web em tempo real. Assim, esse ambiente pode ser usado para criar servidores escaláveis sem usar threading, utilizando então um modelo simplificado de programação orientada a eventos que, por sua vez, faz uso de callbacks para sinalizar a conclusão de uma tarefa (BROWN, 2014).

2.5 Flutter

Flutter é um framework de código aberto que tem permite o desenvolvimento de softwares de interfaces gráficas de usuário para diferentes plataformas utilizando uma mesma base de código, escrita na linguagem Dart. Até o momento de escrita deste trabalho, as plataformas suportadas são Android, iOS, Linux, Mac, Windows, Google Fuchsia e navegadores Web. (WINDMILL; RISCHPATER, 2020).

A linguagem Dart, que é orientada a objetos, fornece a estrutura sintática para o desenvolvimento da modelagem do software, enquanto que várias bibliotecas embutidas no framework Flutter dão ao desenvolvedor acesso a recursos das plataformas, como Bluetooth, WiFi, sistemas de arquivos, entre outros. Os componentes visuais são baseados em classes do tipo Widget, que inclusive pode replicar aspectos visuais das linguagens de design criadas para o iOS e para o Android (linguagem Material Design, da Google) (NAPOLI, 2019).

2.6 Microcontroladores e SoCs

De acordo com Badaway (2003), com avanço das tecnologias de circuito integrado, está sendo possível inserir cada vez mais portas lógicas em um único chip. Como todos os componentes de um sistema computacional são construídos com bases em portas lógicas, é possível construir verdades sistemas num único chip (System-on-chip - SoC), com estes possuindo possivelmente componentes como memória, CPU, processadores digitais de sinais, codificadores/decodificadores, entre outros.

De acordo com a literatura, os SoCs têm uma aquitetura com uma lógica inversa da de PCs comuns, que possuem placas-mãe que fazem a intercomunicação entre os chips. Enquanto um SoC possui todos em apenas um chip todos os componentes necessários para o funcionamento proposto, um PC baseado em placa-mãe precisa que cada componente seja estruturado na forma de um chip, para que a placa-mãe os conecte através de um circuito de interface central. Os SoCs são muito comuns em computação móvel (como em smartphones e tablets), sistemas embarcados e mercados de computação de borda (edge computing) (FLYNN; LUK, 2011).

Um exemplo de dispositivo que utiliza um SoC é a placa de desenvolvimento DOIT Esp32 DevKit v1, que faz uso do SoC ESP32, o qual suporta tecnologias como Wi-Fi, Bluetooth e Ethernet) no mesmo chip. Essa placa é muito utilizada para aplicações experimentais e de aprendizado relacionadas a automação e microcontroladores, podendo ser programada através de vários ambientes distintos, o que inclui a IDE do Arduino (CAMERON, 2021).

2.7 Sinal Eletromiográfico

De acordo com Sadikoglu, Kavalcioglu e Dagman (2017), eletromiografia (EMG) é uma técnica para avaliar a atividade elétrica produzida por músculos esqueléticos quando suas células são eletricamente (por fatores externos ao corpo, por exemplo) ou neurologicamente (por atividades do próprio corpo - voluntárias ou não) ativadas. É medida a diferença de potencial (variando entre 50 V e 30 mV, aproximadamente) de uma parte de um músculo em relação a outra ao decorrer do tempo durante a execução de algum movimento ou com o membro em questão parado.

Os sinais gerados por eletromiografia possuem diversas aplicações clínicas e biomédicas. Alguns exemplos de aplicações clínicas são a avaliação dos sinais para detecções de anomalias, causas de dores musculares, atrofias, etc. Já aplicações biomédicas podem ser a utilização dos mesmos para interação homem-máquina, construção de próteses, entre outras (RAEZ; HUSSAIN; MOHD-YASIN, 2006).

Por fim, ainda de acordo com Raez, Hussain e Mohd-Yasin (2006), existem duas formas de captação EMG: superficial e intramuscular. A EMG superficial (ou sEMG) utiliza eletrodos conectados à parte da pele que fica acima do músculo a ser avaliado. Mais de um eletrodo é necessário porque o que é medido é a diferença de potencial entre o local de aplicação de um eletrodo e o local de aplicação do outro. Um terceiro eletrodo também é recomendado em dispositivos de captação menos precisos para que ele seja um ponto de referência (terra) em relação aos outros, devendo este ser aplicado numa região do corpo não afetada (ou afetada minimamente) pela movimentação do músculo a ser avaliado. A sEMG é uma abordagem menos invasiva para o paciente, porém é mais limitada em relação à intramuscular pelo fato do contato dos eletrodos se restringirem aos músculos superficiais, também por ser influenciada pela profundidade do tecido subcutâneo no local da gravação, além de não ser muito precisa na discriminação entre as descargas de músculos adjacentes ao músculo avaliado. Já a EMG intramuscular consiste na utilização de agulhas para acessar diretamente os músculos a serem avaliados, possuindo uma precisão muito maior que a sEMG, porém com uma aplicação muito mais invasiva e delicada.

3

Trabalhos Relacionados

Neste capítulo serão analisados trabalhos que possuem relação com o trabalho proposto neste texto, assim como serão expostos os resultados de uma pesquisa de opinião feita com possíveis interessados nos resultados obtidos neste trabalho. Os tipos de trabalhos avaliados serão separados em dois: literaturas de cunho acadêmico, como artigos e possivelmente capítulos de livros; e produtos disponíveis no mercado.

3.1 Revisão da Literatura

Para analisar trabalhos acadêmicos, foi utilizado o procedimento de revisão sistemática apresentado por Kitchenham (2004). Baseados em guias consolidados de revisão sistemática na área médica, este procedimento tem como objetivo planejar e executar de forma adequada uma revisão sistemática para trabalhos na área de engenharia de software.

Uma revisão sistemática da literatura (ou somente "revisão sistemática") consiste num tipo de estudo secundário que tem como objetivo responder a um questionamento específico de uma pesquisa de maneira objetiva, confiável e averiguável e até certo ponto reprodutível, através de trabalhos que se mostrem relevantes ao tema desejado.

Sendo assim, o protocolo utilizado constituiu-se das seguintes etapas: definição da questão a ser avaliada na pesquisa; definição da estratégia de busca a ser usada, consistindo esta na escolha das bases de dados onde se fazer a busca, assim como nos termos de busca, palavras-chave e strings disponíveis nos meios e bases de dados utilizados; definição de critérios inclusivos e exclusivos de filtragem para que apenas os trabalhos considerados verdadeiramente relevantes sejam avaliados; e por fim a execução desse planejamento, assim como a exposição dos resultados da mesma.

3.1.1 Questionamento da Pesquisa

Como primeira etapa do planejamento do protocolo, com o intuito de definir claramente o objetivo da revisão, foi definido o seguinte questionamento a ser respondido por este estudo secundário: "*Quais as principais soluções recentes, no âmbito das pesquisas científicas, para monitorar a utilização de músculos através de dispositivos vestíveis?*"

3.1.2 Critérios de Filtragem

Com o objetivo de restringir a quantidade de trabalhos analisados, priorizando suas relevâncias para com este trabalho, foram definidos critérios de inclusão e exclusão para esclarecer quais tipos de trabalhos devem ser ou não considerados na revisão.

Os critérios de inclusão consistem em:

- Estudos primários para o desenvolvimento de dispositivos vestíveis (wearables) relativos à saúde;
- Estudos secundários sobre trabalhos com wearables na área de saúde;
- Trabalhos com sistemas embarcados relativos a procedimentos com eletromiografia, eletrocardiografia ou eletroencefalografia;
- Trabalhos com dispositivos vestíveis que utilizem microcontroladores ou plataformas afins, como Arduino e Raspeberry Pi no desenvolvimento;
- Trabalhos que utilizem plataforma móveis como principal meio de uso, principalmente smartphones e smartwatches;
- Trabalhos que utilizem bancos de dados para armazenar sinais elétricos fisiológicos (EMG, ECG, etc.).

Em paralelo, os critérios de exclusão compreendem:

- Trabalhos que não têm relação com saúde, exercícios físicos ou fisioterapia;
- Trabalhos que não envolvem soluções com tecnologia da informação;
- Trabalhos anteriores a 2013;
- Soluções sem uso de sensores ou sem comunicação com dispositivos externos com sensores;
- Trabalhos que envolvam puramente o acompanhamento de dietas.

3.1.3 Estratégia de Busca

Para encontrar trabalhos que apresentem características que obedeçam os critérios escolhidos para a revisão, é importante decidir quais as bases de dados onde procurar os trabalhos e quais os termos utilizados nas buscas. As bases escolhidas estão entre as mais prestigiadas em termos de trabalhos científicos e foram elas a Engineering Village, a IEEE Xplore, a Scopus e a Web of Science. Para definir os termos e strings utilizadas na busca em cada um desses portais, foi necessário primeiramente definir as palavras-chave que caracterizam o teor da pesquisa e, por conseguinte, uma string genérica, que será utilizada nas buscas em todas as bases de dados, que caracterize de forma adequada o uso das palavras-chave escolhidas.

Sendo assim, as palavras-chave escolhidas foram: “application”, “software”, “mobile application”, “smartphone application”, “android application”, “ios application”, “smartwatch application”, “api”, “database”, “wearable”, “embedded system”, “arduino”, “raspberry pi”, “microcontroller”, “electromyography”, “emg”, “electrocardiography” e “ecg”.

Como consequência, agrupando as que forem consideradas como sinônimos dentro do âmbito deste trabalho, a string genérica gerada a partir dessas palavras-chave foi:

(“software” OR “mobile application” OR “smartphone application” OR “android application” OR “ios application” OR “smartwatch application” OR “api” OR “database”) AND (“wearable” OR “embedded system” OR “arduino” OR “raspberry pi” OR “microcontroller”) AND (“electromyography” OR “emg” OR “electrocardiography” OR “ecg”)

3.1.4 Resultados

Nessa parte do trabalho são apresentados os resumos de cada trabalho analisado, assim como uma análise das características dos mesmos. E por fim, é apresentada uma tabela comparativa que resume as caracteríticas em comum a cada trabalho.

Com o objetivo de analisar a utilização de músculos e batimentos cardíacos durante uma atividade de recuperação muscular, Farjadian, Sivak e Mavroidis (2013) desenvolveram uma camisa inteligente para fornecer feedbacks ao usuário através de sensores de vibração e de um aplicativo de smartphone conectado à camisa. Para tanto, os autores utilizaram sensores EMG e ECG para a captação dos sinais, enviando-os para um Mini Arduino Pro (com microcontrolador ATMEGA 168) com transmissão, através de um módulo Bluetooth, para um smartphone. Todo este aparato, exceto o smartphone, foi acoplado à camisa da maneira mais adequada encontrada. Na aplicação presente no smartphone havia funcionalidades como contagem de repetições, tempo de duração do exercício e outras informações do treino pré-cadastrado. Além disso, as informações do sinal foram enviadas do smartphone a um banco de dados online, o qual poderia ser acessado através de uma aplicação web pelo profissional que estivesse acompanhando as atividades do usuário .

Já Lee et al. (2015) desenvolveram um sistema para controlar o aplicativo específico de

smartphone através de sinais musculares. Os sinais foram captados por sensores superficiais de EMG e transmitidos para um módulo FPGA embutido numa placa DE0-nano. Através de um algoritmo de reconhecimento de padrões e classificação desenvolvido no trabalho, foi definido na placa DE0-nano um mapeamento entre determinados padrões de sinal e eventos a serem disparados para o aplicativo. A partir daí, o tipo de sinal gerado foi transmitido via módulo bluetooth (acoplado à placa) para o smartphone, onde estava instalada uma aplicação para interpretar o evento gerado e executar o comando correspondente ao evento em questão.

Biagetti et al. (2016) elaboraram um sistema robusto de captação e transmissão de sinais fisiológicos, dentre eles EMG e ECG. Foram desenvolvidos circuitos customizados de captação aplicados no corpo do usuário através de eletrodos. Os nós transmitiam o sinal captado para uma estação base receptora através do padrão de camada física IEEE802.15.4 e um protocolo adicional customizado. Desta estação, os dados foram enviados via USB para um computador do tipo PC onde os sinais eram interpretados. O computador também foi útil como servidor HTTP que recebeu requisições de uma aplicação Android.

Mais focado na implementação dos algoritmos, Richer et al. (2014) desenvolveram uma aplicação para dispositivos Android que recebia sinais de um ou mais nós de um sensor Shimmer Realtime Technologies Ltd. que captava sinais ECG e EMG a uma taxa de 256.4 Hz. A aplicação calculou em tempo real a taxa de batimentos cardíacos e cadência nos sinais de EMG através de simplificações de algoritmos de processamentos destes tipos de sinais obtidos na literatura, uma vez que se pretendia economizar tempo de processamento por se tratar de uma aplicação mobile. Como a aplicação é focada em corridas de bicicleta, há funcionalidades adicionais como mapeamento e armazenamento das corridas por GPS (Global Positioning System).

Lee et al. (2017) desenvolveram um sistema de análise emocional de motoristas enquanto dirigiam em cenários diferentes. Para isso, foram captados sinais de EMG e PPG (photoplethysmography) e sinais de movimento da cabeça através de sensores de movimento inercial. Os sinais captados pelos sensores foram transmitidos para um exemplar da plataforma Adafruit FLORA (baseado no microcontrolador Atmega32u4) onde eram convertidos para o formato digital e enviados, via tecnologia Bluetooth BLEW 4.0, para um smartphone com sistema operacional Android. A classificação dos sinais por algoritmos que detectam se o usuário está relaxado, fatigado ou estressado enquanto dirige foi feita a partir de aparelhos celulares.

Lloret et al. (2017) focaram no desenvolvimento de uma arquitetura eficiente em termos de velocidade de transmissão de dados para um sistema que verifica anomalias em pacientes através de dispositivos vestíveis, captando sinais de batimentos cardíacos, taxa de respiração, e pressão sanguínea, por exemplo. Utilizando um smartphone, esses sinais eram enviados para um servidor a fim de compará-los com informações processadas a partir de um banco de dados contendo parâmetros do que é anômalo ou não. Caso houvesse alguma anomalia nos sinais vitais do paciente, um sinal de alerta era enviado para um médico utilizando a mesma aplicação. Através do sistema desenvolvido, o médico também poderia receber um alerta enviado manualmente

pelo paciente. A verificação da anomalia era realizada a partir de um sistema com inteligência artificial baseada em redes neurais, que faz a análise de vários dados (Big Data) captados de servidores de diferentes hospitais. Um dos principais focos do trabalho foi estabelecer que a conexão entre o smartphone e o servidor fosse através de uma rede 5G, devido a sua baixa perda de pacotes e à sua maior velocidade, quando comparada a outras tecnologias móveis.

Utilizando cinco dispositivos MTx, da XSens Inc., para captar alguns sinais vitais provindos do acelerômetro, giroscópio e magnetômetro deste dispositivo, Yurtman e Barshan (2014) verificaram se os movimentos realizados pelo usuário estão corretos com o intuito de tornar a execução dos mesmos mais independente do acompanhamento contínuo de um profissional de fisioterapia ou educação física. Para tanto, inicialmente, e com a ajuda de um profissional, o sistema foi calibrado e os exercícios executados corretamente. Os dados gerados por esses movimentos iniciais foram então configurados como o padrão correto a ser seguido. A partir daí, todos os movimentos executados tiveram seus sinais gerados comparados com os dados padrão para determinar se os mesmos estavam corretos ou não. Esse comparativo foi realizado por meio de um algoritmo denominado Multi-template multi-match DTW que, por sua vez, é baseado no algoritmo DTW (Dynamic Time Warping). O Multi-template multi-match DTW foi executado num laptop com processador quad-core processor de 2 GHz (Intel Core i7 2630QM) e com 8GB de memória RAM, executando o software MATLAB, versão de 32 bits.

Apresentando um trabalho secundário, Hakonen, Piitulainen e Visala (2015) analisaram o estado da arte na utilização e análise de sinais de EMG em relação a diversos aspectos. Inicialmente, foram analisadas as possibilidades na aquisição e pré-processamento do sinal, como a escolha dos tipos, quantidade e posicionamento dos eletrodos, parâmetros da filtragem (para eliminar ruídos) e taxa de amostragem, além de algoritmos de pré-processamento para facilitar análises posteriores. Em seguida, foram explicadas as possíveis técnicas de decodificação das informações contidas no sinal, mostrando qual era mais apropriada para cada tipo de informação desejada. Por fim, o trabalho analisou as tendências futuras para área, assim como aplicações atuais e em potencial.

Em outro trabalho secundário, King et al. (2017) explicaram as diversas técnicas (incluindo algoritmos e arquiteturas) utilizadas na literatura para a análise conjunta de diversos sinais biológicos captados por sensores ou dispositivos vestíveis. Essa análise conjunta é chamada de fusão de dados e serve para auxiliar na compreensão da relação entre os dados e quais tipos de informação eles conseguem fornecer, principalmente quando analisados em conjunto. A revisão também abordou as diversas aplicações da fusão de dados no monitoramento de aspectos relacionados à saúde do usuário, como reconhecimento de atividades, detecção e predição de quedas e monitoramento fisiológico.

Puente, Úbeda e Torres (2017) apresentaram uma metodologia de ensino de instrumentação voltada para sinais biomédicos através de práticas de laboratório com projetos envolvendo Arduino e com o shield de sensores chamado e-Health, que pode captar diversos sinais, como

EMG, ECG, posição e temperatura do corpo e pressão sanguínea, por exemplo. Apesar de não apresentar o desenvolvimento de uma solução em si, o artigo contribuiu para o esclarecimento de quais passos tomar para o aprendizado das ferramentas a serem utilizadas em qualquer tipo de trabalho relacionado à análise de sinais biológicos.

Na revisão de Athavale e Krishnan (2017) foi avaliado de maneira holística a criação de dispositivos vestíveis e suas possibilidade de impacto no mercado de saúde. Os autores fizeram uma análise de dispositivos e técnicas de captação existentes no mercado até a data de publicação do trabalho, assim como uma comparação entre a qualidade da análise proporcionada por tais dispositivos em comparação com medições clínicas, feitas com equipamentos mais robustos e tradicionais. Dentre as diversas conclusões obtidas, destacam-se: i. os dispositivos possuem uma substituição adequada para equipamentos tradicionais (contanto que se utilizem técnicas e algoritmos para captação das partes que realmente relevantes do sinal); ii. o sinal de EMG possui uma complexidade suficientemente superada pelas tecnologias de análise e dispositivos móveis existentes atualmente.

Com o intuito de provocar estímulos usados na recuperação de músculos da maneira mais adequada possível, Wang et al. (2017) desenvolveram um sistema que provoca estímulos através de um conjunto de eletrodos distribuídos em formato de matriz (Multi-pad Electrode), captando os movimentos gerados por eles através de uma luva inteligente para que novos estímulos sejam criados mais apropriadamente às respostas obtidas. Essa luva era composta por uma placa Intel® Edison Compute Module que possui um núcleo dual AtomTM core de 500 MHz e um microcontrolador QuarkTM de 100 MHz com sistema operacional Yocto Linux v1.6. A configuração inicial foi feita através de um aplicativo Android que enviava um sinal por tecnologia Wi-fi para um módulo de controlador de estímulos construído com circuitos customizados em placas de circuito impresso e conectados por cabos de circuito flexível impressos (Flexible Printed Circuit - FPC). Este último se conectava por Wi-fi a um módulo controlador de eletrodos (constituído principalmente de um microcontrolador ARM® Cortex®-M3, um módulo Wi-fi UART HLK-M30, da Hi-Link Electronic Inc., e um circuito seletor de eletrodos), cuja função era decodificar o sinal recebido para ativar de forma adequada a matriz de eletrodos estimulantes.

A ferramenta GaitTool App foi desenvolvida por Plewa et al. (2017) com o intuito de ajudar na recuperação e correção de caminhada por meio de leituras de sinais de mecanomiografia (MMG) de músculos da perna (músculo tibial anterior e músculo gastrocnêmio), bem como da análise desses sinais e da transmissão de biofeedbacks sonoros positivos ou negativos, a depender da qualidade do padrão de caminhada. A leitura dos sinais foi realizada com acelerômetros triaxiais ADXL337, da Analog Devices Inc., transmitido via cabos para uma placa Arduino UNO MCU que processou o sinal em linguagem C e transmitiu os resultados via Bluetooth (por dispositvos BLE Mini) para um aplicativo Android. Esse dispositivo fez análises mais detalhadas, além de armazenar as informações e gerar o som, tipo de biofeedback escolhido para aplicação.

Pancholi e Joshi (2018) projetaram um módulo portátil de aquisição e transmissão sem

fio de sinais de EMG para um notebook com aplicação voltada para a área de desenvolvimento de próteses. O módulo de aquisição foi composto por oito eletrodos como canais de captação, um circuito específico de proteção contra ruídos e picos de tensão, assim como circuito integrado ADS1298 para as operações de aquisição do sinal e um processador ARM cortex série M4, onde foi feito o processamento do sinal. O sinal foi transmitido via módulo Bluetooth numa banda ISM de 2,4 GHz de frequência para um laptop contendo uma aplicação Java para análise do sinal, tentando identificar os padrões de movimento através de algoritmos de aprendizado de máquina. Os autores testaram diferentes algoritmos e compararam o desempenho dos mesmos em relação a diferentes fatores. Por fim, foi feito um outro comparativo em relação ao preço de implementação frente a outras soluções presentes no mercado, concluindo-se que a opção estudada por Pancholi e Joshi era mais barata dentre as alternativas existentes.

Benatti et al. (2015) focaram no reconhecimento de gestos em tempo real através da captura de sinais de EMG, desenvolvendo tanto o hardware quanto o software necessário. O hardware foi composto por eletrodos superficiais conectados a um dispositivo de aquisição do sinal (Cerebro ASIC, um circuito projetado pelos próprios autores num trabalho anterior), que se comunicava via SPI com um microcontrolador ARM Cortex M4, onde foram armazenados modelos pré-definidos de reconhecimento de gestos utilizando a técnica SVM (Support Vector Machine). Os resultados foram armazenados num cartão SD via interface SPI ou transmitidos sem fio via tecnologia Bluetooth padrão 2.1.

Por outro lado, Brunelli et al. (2016) trabalharam na durabilidade e economia de energia da plataforma desenvolvida, analisando diversas opções de tecnologia para a aquisição e processamento de reconhecimento de gestos. A captação foi feita por eletrodos acoplados em sensores Ottobock 13E200, que transmitiu o sinal pré-processado para um microcontrolador ARM Cortex M3, onde foi feito o processamento do sinal e a partir do qual o sinal foi transmitido via protocolo Bluetooth Low Energy (BLE) ou Low Power WiFi (LpWiFi) para um smartphone Android, o qual serviu de interface entre o sistema de captação e a prótese a ser controlada possivelmente através das mesmas opções de interface de rádio citadas acima.

Al-Kababji et al. (2019), por sua vez, desenvolveram um sistema de monitoramento de idosos em casos de quedas. Para isso, foram captados sinais de ECG e de um acelerômetro através do dispositivo de captação ODROID-XU4, no qual foram implementados algoritmos de aprendizado de máquina capazes de detectar possíveis quedas de idosos monitorados. Identificada a queda, uma requisição via WiFi era enviada a uma API REST que fazia a interface com o banco de dados Firebase Realtime, responsável por notificar uma aplicação móvel Android sobre o evento. Essa aplicação móvel com uma interface gráfica tinha o propósito de estar nas mãos de familiares ou pessoa responsável pelo idoso que estava sendo monitorado a fim de notificá-los sobre o ocorrido.

De forma similar, Rodriguez e Julcapoma (2020) desenvolveram um sistema de monitoramento de diferentes sinais vitais, como ECG, EMG e temperatura corporal, com foco em como

transmitir tais sinais em tempo real para diferentes aplicações consumidoras. Para a captação do sinal foi usado sensores de captação correspondentes ao tipo de sinal, acoplados, cada um, a um módulo ESP8266, que transmitia um sinal via WiFi, pelo protocolo WebSocket, para um servidor na nuvem implementado em Node.js. Este último, também via WebSocket, transmitia o sinal em tempo real para uma aplicação web, cujo objetivo era imprimir um gráfico da intensidade do sinal em relação ao tempo.

Shaown et al. (2019) implementaram um sistema de monitoramento, armazenamento e transmissão de sinais de ECG. A captação foi feita a partir de um sensor EMG acoplado a um microcontrolador. Os níveis de intensidade do sinal eram exibidos num módulo LCD e armazenados num cartão SD, ambos conectados ao microcontrolador. Além disso, via módulo WiFi, o sinal era enviado para um servidor na nuvem, responsável por realizar uma transmissão em tempo real para uma aplicação web.

	Possui aplicação móvel (Android ou iOS) ou Web	Processa EMG	Possui API de consumo do sinal	Usou armazenamento de dados permanente	Usa banco de dados de temporal	Transmissão sem fio dos sinais
Farjadian, Sivak e Mavroidis (2013)	x	x		x		x
Lee et al. (2015)	x	x				
Biagetti et al. (2016)	x	x		x		
Richer et al. (2014)	x	x				
Lee et al. (2017)	x	x				x
Lloret et al. (2017)	x	x	x			x
Yurtman e Barshan (2014)						
Hakonen, Piitulainen e Visala (2015)		x				
King et al. (2017)						
Puente, Úbeda e Torres (2017)		x				
Athavale e Krishnan (2017)						
Wang et al. (2017)	x					x
Plewa et al. (2017)	x					x
Pancholi e Joshi (2018)	x					x
Benatti et al. (2015)		x				x
Brunelli et al. (2016)	x					x
Al-Kababji et al. (2019)	x			x		x
Rodriguez e Julcapoma (2020)	x		x			x
Shaown et al. (2019)			x	x		x

Tabela 1 – Comparação de características dos trabalhos avaliados

4

Proposta e Implementação do Sistema MyoVita

Na análise dos trabalhos relacionados foi observado que nenhum documento envolveu o armazenamento de sinais vitais a longo prazo em bancos de dados do tipo série temporal, o qual, conforme discutido anteriormente na seção de fundamentação teórica, é bastante apropriado para dados de grande volume em curto intervalo de tempo, a exemplo das amostras de eletromiografia. Além disso, os softwares desenvolvidos nos trabalhos relacionados possuíam, de modo geral, pelo menos duas das seguintes entidades: captação, armazenamento e amostra gráfica de sinais. Essas três entidades geralmente são implementadas de maneira bastante acoplada, sem que haja a possibilidade de rodá-las em instâncias de execução e/ou em máquinas diferentes. Esse alto acoplamento de, no mínimo, dois desses componentes limita algumas possíveis funcionalidades do software aos recursos da máquina na qual os componentes são executados. Para ilustrar essa situação, toma-se o caso de um sistema embarcado, que é uma opção viável para implementação de um componente de captação, por trazer menor custo e flexibilidade, mas pode não ser o mais apropriado para execução de algoritmos complexos de detecção de padrões que exigem mais poder computacional, além de não ser um tipo de software criado para armazenamento robusto e a longo prazo de um grande volume de informações.

A proposta do presente trabalho pretende sanar essas duas limitações encontradas nos trabalhos avaliados, desenvolvendo um sistema robusto de armazenamento de sinais de EMG através do uso de um banco de dados de série temporal, implementando de forma pouco acoplada essa entidade de armazenamento em relação às entidades de captação e disponibilização do sinal para um usuário final, permitindo que as três entidades possam ser executadas de forma independente.

4.1 Estrutura Geral da Proposta

O ponto central da proposta consiste em um software principal com interface para recebimento de sinais EMG através do protocolo MQTT, assim como uma interface HTTP com padrão REST para que aplicações possam consumir os dados de usuários e seus sinais vitais. Uma interface via protocolo WebSocket também é fornecida para transmissão em tempo real dos sinais que estão sendo recebidos a fim de que eles sejam transmitidos para uma aplicação consumidora em tempo real.

Com o propósito de ilustrar a utilização do software principal, foram construídos dois softwares adicionais. O primeiro software construído capta o sinal EMG e envia para o servidor, enquanto o segundo consome os dados e amostras de sinal muscular considerando dados em tempo real e históricos do usuário em questão num determinado intervalo de datas. Para o caso de utilização real, o software principal pode ser utilizado para diversas aplicações que almejam consumir os dados ou captá-los da forma desejada, abrindo, assim, margem para os dados serem exibidos em diversas possibilidades de interface de usuário, bem como para vários tipos de sinal serem captados da maneira mais eficiente a ser estudada. Na Figura 5 encontra-se uma ilustração da ideia geral dos softwares construídos e da comunicação entre eles.

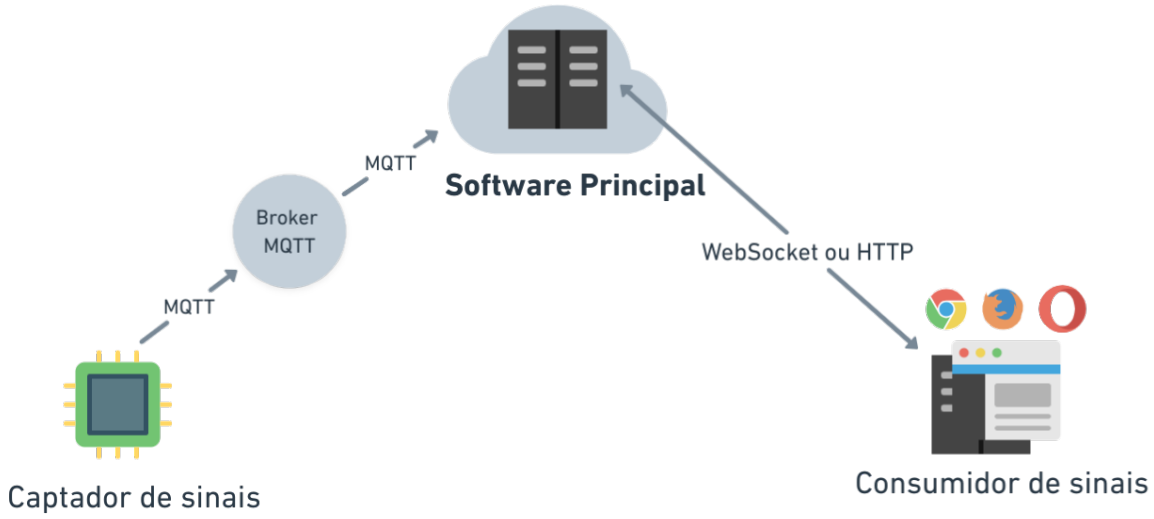

Figura 5 – Diagrama da estrutura geral dos softwares construídos

4.2 Software Principal - Servidor de EMG

A aplicação principal foi construída na linguagem JavaScript, utilizando Node.js como ambiente de interpretação da linguagem, a biblioteca express.js para lidar com rotas de requisições HTTP, assim como uma biblioteca chamada mqtt.js para escutar recebimentos de dados de sinais via protocolo MQTT.

No tocante ao recebimento de sinais EMG, ao receber uma mensagem com o tópico chamado “lpctcc/emg” no broker cujo endereço é “mqtt://broker.hivemq.com” via protocolo MQTT, o software proposto armazena o sinal no banco de dados InfluxDB através da biblioteca influxdb-client. O modelo de dados recebido via MQTT se encontra no formato JSON, contendo as propriedades “id_user” e “samples”, essa última sendo um array das amostrados do sinal gerado. O protocolo MQTT foi escolhido para o recebimento dos sinais porque os sinais precisam ser captados por sensores que são geralmente acoplados a sistemas embarcados para o processamento do mesmo. Como dito na seção de fundamentação teórica, o MQTT é mais leve do que protocolos como HTTP e WebSocket e, portanto, mais apropriado para softwares embarcados.

Ao receber esse formato de dados, a aplicação principal verifica se no banco relacional MySQL existe um usuário cadastrado com o id_user passado. Em caso positivo, as amostras são armazenadas no banco temporal, porém, em caso negativo, as amostras não são armazenadas. Não foi planejado nenhum mecanismo para comunicar esses casos negativos ao software que enviou os sinais. Além disso, caso exista uma conexão com WebSocket aberta referenciando o id_user recebido via MQTT, a aplicação principal enviará as amostras para o cliente desta conexão WebSocket.

O trecho do código correspondente ao processamento feito ao receber o sinal via MQTT pode ser encontrado na Figura 6.

Conforme observado através de sua URL, o broker selecionado, denominado HiveMQ, é gratuito e está disponível publicamente, já que o desenvolvimento do presente trabalho foi de cunho experimental, sem uso comercial e sem restrições de segurança com dados de usuários.

O trecho do código correspondente a uma parte da configuração da conexão por WebSocket pode ser encontrado na Figura 7.

Além do formato via WebSocket mencionado, também é possível consumir os dados armazenados através de uma API REST disponibilizada. Os endpoints disponíveis são os seguintes:

- GET /users - retorna a lista de usuários cadastrados e as informações básicas dos mesmos, que são id_user, nome e idade;
- GET /users/:id_user - retorna as informações básicas do usuário (id_user, nome e idade) cujo id_user é o “id” passado como parâmetro;
- GET /users/:id_user/samples?from=date_on_utc_format&to=date_on_utc_format - retorna as amostras do tipo EMG daquele usuário que foram registradas no intervalo de horas passado como parâmetros de query;
- POST /users - recebe no corpo da requisição dados a serem armazenados de um novo usuário.

```
async function storeSignalSample(idUser, signalType, signalLevel) {
  try {
    const client = new InfluxDB({
      url: INFLUX_DB_URL,
      token: INFLUX_DB_TOKEN,
    });

    const writeApi = client.getWriteApi(
      INFLUX_DB_ORGANIZATION,
      INFLUX_DB_BUCKET
    );
    writeApi.useDefaultTags({ host: INFLUX_DB_HOST });
    const point = new Point(signalType)
      .floatField("signal_level", 1 + signalLevel)
      .timestamp(new Date())
      .tag("id_user", idUser);

    writeApi.writePoint(point);

    await writeApi.close();
    console.log("FINISHED");
  } catch (error) {
    console.error(error);
    console.log("\\nFinished ERROR");
  }
}

async function processSignal(signal) {
  const { idUser, signalSamples } = JSON.parse(signal);

  for (const signalSample of signalSamples) {
    await storeSignalSample(idUser, "emg", signalSample);
  }
  const [[{ userExists }]] = await relationalDatabaseConnection.execute(
    `SELECT count(*) as userExists  from user where id_user = ${idUser}`
  );

  if (userExists !== 1) return 0;

  if (webSocketClients.length > 0) {
    webSocketClients[0].sendUTF(signalSamples);      You, seconds ago • U
    console.log("Transmitted to websocket");
  }
}
```

Figura 6 – Trecho do código do software principal correspondente ao processamento feito ao receber o sinal via MQTT.

O trecho do código correspondente a parte da API REST pode ser encontrado na Figura 8.

O código completo do software principal pode ser encontrado nesse link.

4.3 Modelos de Armazenamento do Software Principal

Essa aplicação possui duas formas de armazenamento de dados, a saber: i. armazenamento dos sinais vitais, mais especificamente, das amostras de EMG do usuário num banco

```
const WebSocketServer = require("websocket").server;
const http = require("http");

const server = http.createServer(function (request, response) {
  console.log(new Date() + " Received request for " + request.url);
  response.writeHead(404);
  response.end();
});
server.listen(8000, function () {
  console.log(new Date() + " Server is listening on port 8000");
});

wsServer = new WebSocketServer({
  httpServer: server,
  autoAcceptConnections: false,
});

wsServer.on("request", function (request) {
  const connection = request.accept("echo-protocol", request.origin);
  console.log(new Date() + " Connection accepted.");
  webSocketClients.push(connection);

  connection.on("message", function (message) {
    console.log(message);
    if (message.type === "utf8") {
      console.log("Received Message: " + message.utf8Data);
      connection.sendUTF(message.utf8Data);
    } else if (message.type === "binary") {$      You, seconds ago • Uncommitted
      console.log(
        "Received Binary Message of " + message.binaryData.length + " bytes"
      );
      connection.sendBytes(message.binaryData);
    }
  });
  connection.on("close", function (reasonCode, description) {
    console.log(
      new Date() + " Peer " + connection.remoteAddress + " disconnected."
    );
  });
});
```

Figura 7 – Trecho do código do software principal correspondente a uma parcela da conexão por WebSocket.

de dados temporal InfluxDB; ii. armazenamento a partir de um banco relacional MySQL para guardar dados de cadastro de usuários.

No modelo de dados de um usuário existe apenas uma tabela no banco relacional MySQL que o representa e a única entidade nesse banco é o usuário por motivos de simplificação do escopo da pesquisa. Para tanto, um banco relacional foi selecionado justamente porque outras informações que possam vir a ser inseridas no futuro serão, provavelmente, melhor modeladas nesse tipo de banco. Em trabalhos futuros, por exemplo, se houver a pretensão de cadastrar profissionais para acompanhar o usuário, pode-se fazer necessário registrar doenças conhecidas e relacioná-las com a entidade usuário para indicar se o mesmo é portador ou não. A Figura 9 ilustra o diagrama de tipagem da tabela usuário.

```
restApi.get("/users", async (httpRequest, httpResponse) => {
  const [users] = await relationalDatabaseConnection.execute(
    `SELECT * from user`
  );
  return httpResponse.json(users);
});

restApi.get("/users/:id_user", async (httpRequest, httpResponse) => {
  const { id_user: idUser } = httpRequest.params;
  const [user] = await relationalDatabaseConnection.execute(
    `SELECT * from user where id_user = ${idUser}`
  );
  return httpResponse.json(user);
});

restApi.post("/users", async (httpRequest, httpResponse) => {
  const { name, age } = httpRequest.body;

  if (!name) {
    return httpResponse.boom.badRequest("A new user must have a name");
  }

  const [{ insertId: newUserId }] =
    await relationalDatabaseConnection.execute(
      `INSERT INTO user (name, age) VALUES ('${name}', ${age ? age : null})`
    );

  const [user] = await relationalDatabaseConnection.execute(
    `SELECT * from user where id_user = ${newUserId}`
  );

  return httpResponse.json(user);
});
```

Figura 8 – Trecho do código do software principal correspondente a uma parte da API REST.

User	
id_user	INT
name	VARCHAR
age	INT

Figura 9 – Tabela User (Usuário)

No modelo de dados das amostras de sinais, que, dentro do escopo deste trabalho utiliza apenas o sinal elétrico muscular (EMG), foi armazenado um ponto no banco InfluxDB, ou seja, um registro, para cada amostra do sinal. Esse ponto, conforme discutido anteriormente (vide seção de fundamentação teórica) possui alguns parâmetros que o caracterizam, como chave do campo, valor do campo, chave de tag e valor de tag. Uma tabela demonstrando alguns pontos preenchidos no banco temporal pode ser vista na Figura 10. Para cada ponto, como chave do campo (coluna "_field"), foi armazenada a string "signal_level", já como valor (coluna

"_value") foi armazenado o valor da intensidade em milivolts (mV) captada daquela amostra do sinal, enquanto que para identificar o usuário ao qual o sinal pertence, foi colocada uma coluna chamada "id_user" (que seria uma chave de tag) onde é inserido o id do usuário no banco relacional MySQL. Isso significa que é através da tag "id_user" no banco temporal que as informações são relacionadas entre o banco de dados relacional e o temporal.

_time	_measurement	id_user	_field	_value
2020-08-18T00:00:00Z	emg	1	signal_level	23.43234543
2020-08-18T00:00:01Z	emg	1	signal_level	22.13293
2020-08-18T00:00:02Z	emg	2	signal_level	27.904543

Figura 10 – Representação tabular de alguns pontos no banco de dados de série temporal

4.4 Software Captador de Sinal

Para demonstrar um exemplo de alimentação do sistema principal, foi desenvolvido um protótipo de dispositivo para captação de um sinal vital específico: o EMG. Para isso, foi utilizado um sensor de sinal eletromiográfico denominado Advancer Technologies's Muscle Sensor v3, o qual necessita de duas baterias de 9V para sua alimentação, assim como um cabo conector de três pinos para viabilizar sua conexão aos eletrodos que captam o sinal muscular. A fim de obter um protótipo menos invasivo na captação do sinal, foi selecionada a captação superficial do sinal muscular através de pads, os quais foram, durante a captação, conectados na pele do usuário no posicionamento indicado pela literatura: dois eletrodos próximos e em cima do músculo a se avaliar, e um terceiro eletrodo numa parte do corpo que não sofre alterações musculares no momento em que o músculo avaliado é utilizado.

Através de uma protoboard, o sensor eletromiográfico foi conectado a uma placa de desenvolvimento DOIT Esp32 DevKit v1 que faz uso do SoC ESP32. Essa conexão foi executada com o intuito de enviar o sinal para o software principal, uma vez que o ESP32 possui suporte a WiFi e é programável através da plataforma Arduino, que possui bibliotecas para utilização do protocolo MQTT para envio e recebimento de dados ou mensagens.

A placa selecionada para o desenvolvimento do presente trabalho foi configurada para utilizar credenciais fixas de uma rede WiFi, de tal modo que só fosse possível alterá-las através da reprogramação da placa com credenciais novas. A mesma lógica foi utilizada para o id_user enviado juntamente com as amostras de sinal para o software principal, ou seja, o id_user de um usuário já conhecidamente presente no banco de dados foi colocado de forma estática na programação da placa para ser enviado por MQTT. Durante o procedimento, por se tratar de

um protótipo e com o intuito de alimentar a placa, a mesma permaneceu sempre conectada ao computador responsável por programar seu software. A conexão entre os pads dos eletrodos no usuário, sensor e a placa de desenvolvimento pode ser visualizada na Figura 11.

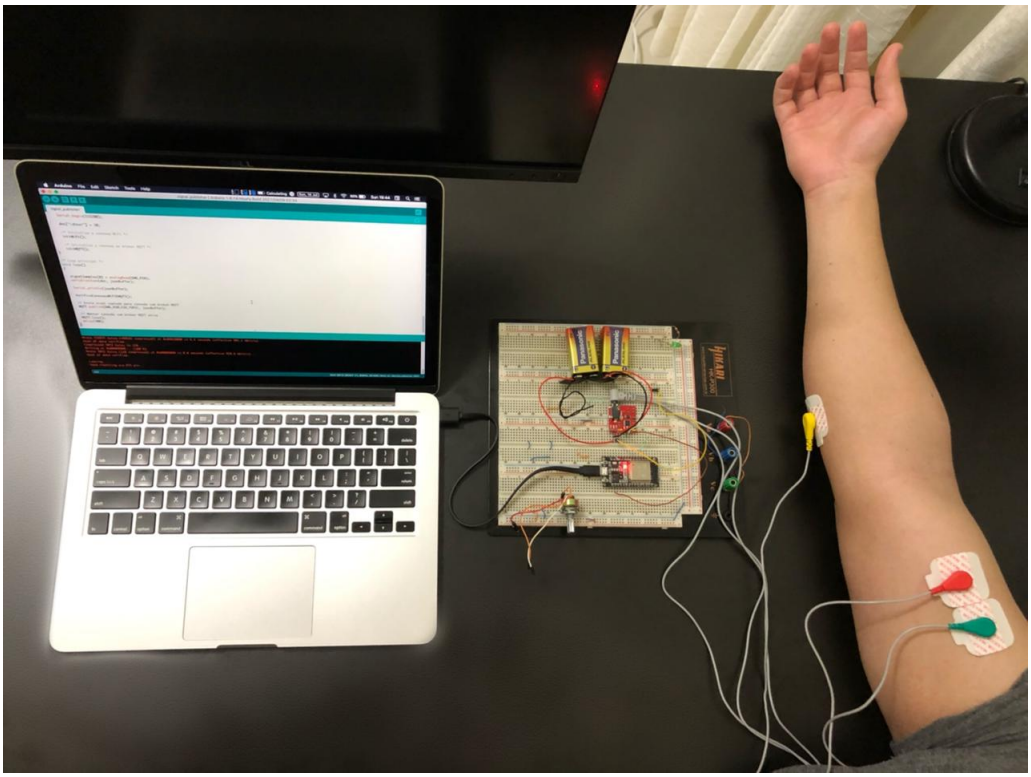

Figura 11 – Configuração do captador de sinal

O trecho principal do código, onde há captação do sinal pelo sensor e envio para o broker MQTT, pode ser encontrado na Figura 12. O código completo do software captador de sinal pode ser encontrado nesse link.

4.5 Software Consumidor de Sinais Vitais

Foi desenvolvida uma aplicação web utilizando o framework Flutter para exibir informações de captação do sinal, como gráficos do sinal EMG em determinado intervalo de tempo ou, até mesmo, em tempo real, assim como para exibir as informações básicas do usuário, como nome e idade. Essa aplicação também permite a criação de um usuário, assim como a seleção do usuário cujo sinal EMG será mostrado na tela de gráficos.

A aplicação inicia numa tela onde é possível ver os usuários cadastrados e selecionar aquele cujos sinais musculares deseja-se visualizar. Nessa mesma tela também há um botão de criação de usuário que, se clicado, leva a uma tela com um formulário de cadastro de novo usuário. Ambas as telas usam o protocolo HTTP para consumir a API REST da aplicação principal para fazer suas operações.

```
/* Função de setup inicial do software */
void setup()
{
    Serial.begin(115200);

    doc["idUser"] = 10;

    /* Inicializa a conexão WiFi */
    initWiFi();

    /* Inicializa a conexão ao broker MQTT */
    initMQTT();
}

/* Loop principal */
void loop()
{

  signalSamples[0] = analogRead(EMG_PIN);
  serializeJson(doc, jsonBuffer);

  Serial.println(jsonBuffer);

  VerificaConexoesWiFIEMQTT();

  // Envia sinal captado para conexão com broket MQTT
  MQTT.publish(EMG_PUBLISH_TOPIC, jsonBuffer);

  // Manter conexão com broker MQTT ativa
  MQTT.loop();
  delay(300);
}
```

Figura 12 – Trecho principal do código do software captador de sinais.

Na lista de usuários, quando se clica num dos usuários, a aplicação abre uma tela onde é mostrado um gráfico de área das amostras EMG do usuário em relação ao tempo, onde o eixo vertical representa a tensão em milivolts (mV) das amostras e o eixo horizontal representa o tempo em segundos (s) de quando a amostras correspondentes foram captadas. Há dois modos para essa tela: o modo “Tempo Real”, que usa o protocolo WebSocket para mostrar o gráfico do sinal produzido aproximadamente no momento de execução por aquele usuário; e o modo “Histórico”, onde há um seletor de intervalo de tempo e que utiliza o protocolo HTTP para consumir a API REST do software principal para buscar as amostras do sinal daquele usuário no intervalo de tempo selecionado. A Figura 13 mostra o protótipo da tela de gráficos no modo “Histórico”, com o seletor de datas.

O código desenvolvido no presente trabalho correspondente ao software consumidor de sinais pode ser encontrado nesse link.

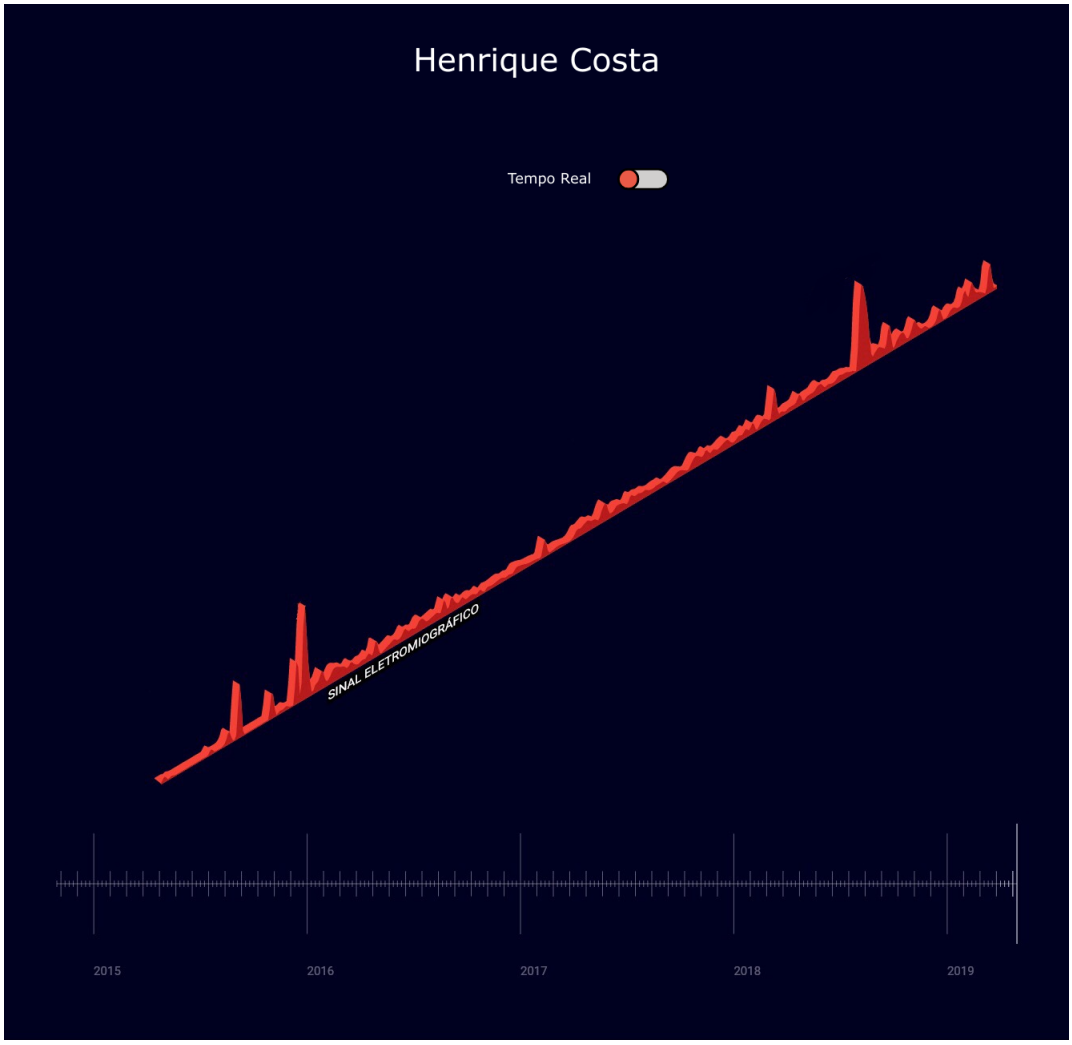

Figura 13 – Protótipo da tela do gráfico de Tensão [mV] x Tempo [s] de sinal eletromiográfico do usuário Henrique Costa na aplicação Web

4.6 Fluxo Completo de um Sinal

O sinal mioelétrico, ao ser captado pelo sensor, é pré-processado no microcontrolador ESP32, onde é feita sua amostragem e envio das amostras através do protocolo MQTT por sinal WiFi para o broker MQTT, que por sua vez envia as amostras ao software principal com o id_user do usuário que está tendo seu sinal vital captado. A aplicação principal recebe os dados enviados pelo captador de sinais, verifica se o id_user recebido é de algum usuário no banco relacional e, caso seja, no banco temporal ele cria um ponto para cada uma das amostras de sinal recebidas. Ao mesmo tempo, na aplicação principal, se houver alguma outra aplicação escutando os sinais de determinado usuário via WebSocket, a aplicação principal envia essas amostras recém-recebidas para o software que está escutando. É o caso do software web consumidor feito em Flutter. No modo "Tempo Real" da aplicação, ele mostra em gráficos de tensão por tempo com as amostras que o software principal acabou de receber do usuário cuja aplicação consumidora está escutando.

Outra possibilidade é visualizar os sinais de EMG passados em determinado intervalo de tempo através da aplicação em Flutter no modo “Histórico”, que consome a API REST do software principal. Supondo que se pretenda visualizar as amostras do usuário de id_user 10 entre 13/07/2021 às 02:00:12 até 16/07/2021 às 13:00:00, é enviada uma requisição HTTP no formato GET users/1/samples?from=2021-07-13T02:00:12.000Zto=2021-07-16T13:00:00.000Z, da aplicação consumidora para o software principal. Este último recebe a requisição, verifica se o id_user existe no banco MySQL e, se existe, faz uma busca no banco temporal neste intervalo de tempo e retorna a resposta para a aplicação consumidora exibir o gráfico do sinal naquele intervalo.

A Figura 14 traz a imagem que representa a interligação entre todos as entidades do sistema MyoVita.

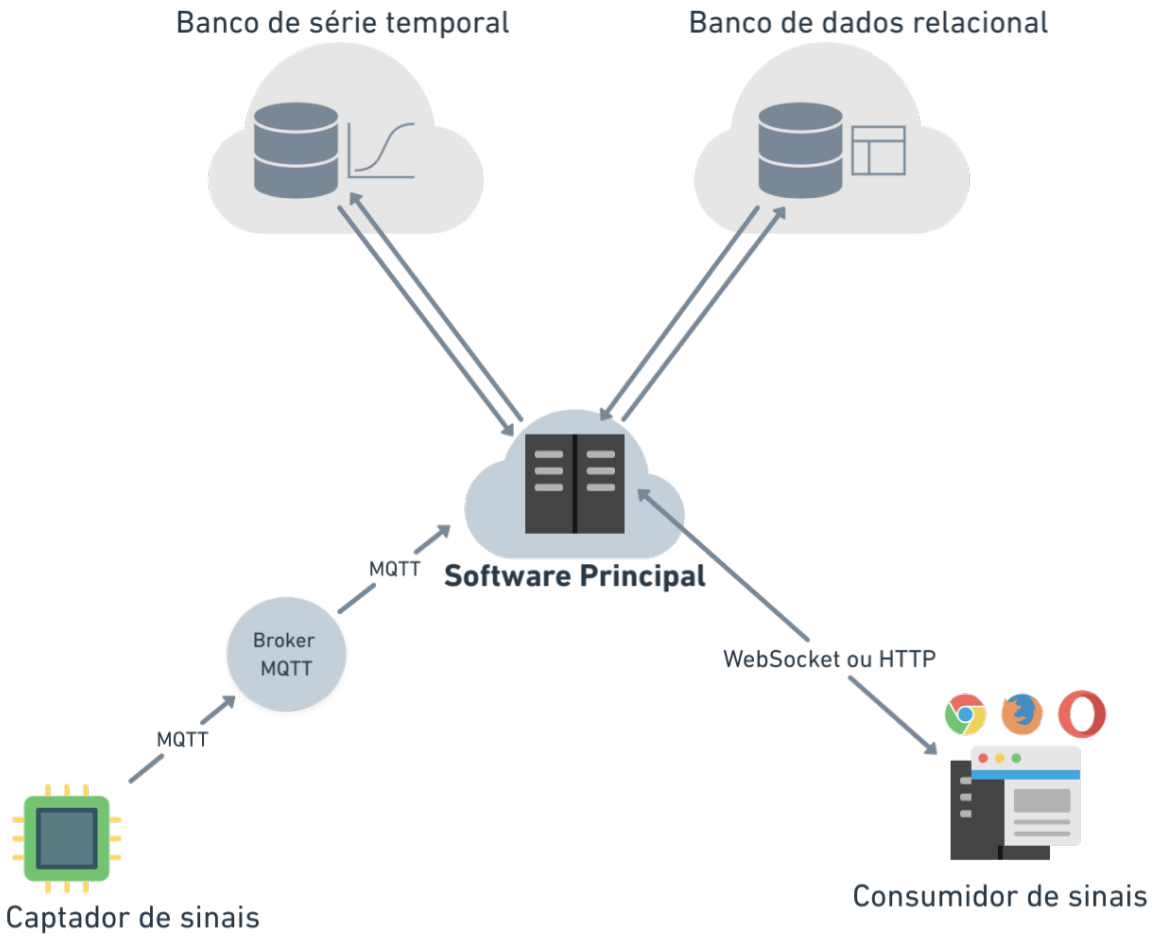

Figura 14 – Representação de fluxo que o sinal muscular percorre pelos softwares

4.7 Resultados

Como o intuito do trabalho é o sistema de armazenamento e transmissão de sinais EMG, ou seja, o software principal, testes funcionais foram feitos nessa aplicação principal de forma individual, isolado dos outros softwares.

No software principal, os componentes testados foram:

- API REST: cadastro de usuário; consumo de lista de usuários existentes; atualização de dados de usuário; e consumo do histórico de sinais EMG de um usuário em determinado intervalo de datas;
- Componente de transmissão online de sinal: recebimento de sinal via MQTT, verificação se usuário enviado realmente existe no banco de dados MySQL. Caso exista, ele deve ser armazenado no banco InfluxDB corretamente com o registro de hora em que chegou no software principal, e também transmitir pras conexões WebSocket que estiverem abertas.

A API REST foi testada através do software Postman, que é um consumidor de APIs via HTTP e outros protocolos. Para verificar toda a execução dos testes, pode-se assistir este vídeo. O teste foi considerado bem sucedido por ter sido possível cadastrar e editar um usuário no banco relacional, listar os dados de um usuário específico, assim como listar os usuários cadastrados no banco relacional.

Para testar o componente de transmissão online de sinal, foram criados dois scripts menores em Node.js para simularem um envio de sinal via MQTT e outro script para simular um cliente WebSocket que se conecta com o software principal e fica escutando sinais do usuário de id_user 10, já conhecidamente armazenado no banco de dados relacional MySQL.

Além disso, para verificar que o sinal enviado foi armazenado com sucesso no banco de série temporal InfluxDB, foi enviada uma requisição para o endpoint GET users/:id_user/samples passando a hora em que foi feito o teste como intervalo inicial (parâmetro *from*).

Para ver a execução dos testes dos componentes de tempo real do software principal, pode-se observar o vídeo neste link. O teste pode ser considerado bem sucedido, por ter validado o funcionamento completo do recebimento, armazenamento e transmissão do sinal em tempo real.

Para testar o software de captação foi verificado se o sinal foi de fato captado corretamente do sensor e enviado via MQTT para o broker. Para isso, são imprimidos os valores captados pelo sensor no terminal de texto da IDE Arduino. Para mais detalhes, pode-se observar o vídeo neste link.

5
Conclusão

Esse trabalho teve como objetivo desenvolver um software de armazenamento permanente de sinais musculares (EMG), assim como implementar softwares de captação e exibição de sinais de forma desacoplada do software de armazenamento. Para entender a melhor abordagem de implementação foi realizada uma revisão sistemática da literatura no formato proposto por Kitchenham (2004), chegando-se à conclusão da necessidade de uma proposta de modelagem de dados específica para o armazenamento robusto e de longo prazo de sinais vitais amostrados.

A abordagem da proposta foi desenvolver um software principal para dar suporte ao armazenamento permanente de sinais temporais via banco de série temporal, assim como dados com diferentes estruturas que estão relacionados a esses sinais, como os dados de um usuário que gerou os sinais. Este software também permitiu comunicações online com softwares geradores de sinal via protocolo MQTT, softwares consumidores de sinal via protocolo WebSocket e consumo do histórico de dados via API REST. O software principal foi feito utilizando a linguagem JavaScript e a biblioteca Node.js como principais ferramentas.

Além do software principal, foi planejada a construção de dois softwares auxiliares para ilustração do uso do software principal. Um deles foi o software captador de sinais, que utilizou uma protoboard, um sensor eletromiográfico e uma placa de desenvolvimento DOIT Esp32 DevKit v1, que faz uso do SoC ESP32 e é programável via IDE Arduino. O outro foi um software Web construído com o framework Flutter para consumir e exibir os sinais musculares em um gráfico de área, a partir do consumo das interfaces WebSocket e API REST da aplicação principal, assim como listar, editar e criar usuários através da mesma API REST.

O objetivo do trabalho foi parcialmente atingido, uma vez que foi possível armazenar amostras de sinais EMG para consultas posteriores de forma bem sucedida e com baixo acoplamento com a aplicação que gerou as amostras. Também foi possível captar amostras reais de um músculo e enviá-la para o software de armazenamento. No entanto, o software de exibição do sinal muscular via interface gráfica não passou da fase de planejamento, prototipação de tela e

início de desenvolvimento do código, tendo sua implementação completa ficado como uma das sugestões para trabalhos futuros.

5.1 Trabalhos Futuros

Todos os três softwares desenvolvidos possuem possibilidades de melhorias e expansões de suas funcionalidades.

No software principal, é possível incrementar os tipos de entidades possíveis no banco relacional. Por exemplo, a adição de uma entidade "profissional", que possuiria relação com vários usuários, para acompanhar os mesmos. Outra entidade possível seria "clínica", que poderia possuir diversos profissionais e usuários (pacientes), expandido assim o escopo organizacional que o software principal poderia englobar. Além da modelagem no banco relacional dessas entidades, endpoints da API REST também poderiam ser criados para permitir o uso dessas mesmas por softwares clientes, como aplicação web.

Ainda no software principal, outros tipos de sinais vitais também poderiam ser suportados. O ECG (eletrocardiograma), que representa os sinais elétricos do coração, em termos de necessidade de armazenamento, possui características muito similares ao EMG por se tratar de um sinal amostrado no tempo quando convertido para o formato digital, podendo assim ser armazenado no banco de dados de série temporal. Além disso, por permitir a existência de um histórico de sinais vitais de um paciente e esse histórico estar presente numa máquina independente do dispositivo de captação, seria possível fazer diversas análises do mesmo em busca de padrões que indiquem doenças ou anomalias, mesmo que os softwares que fizerem essas análises demorem ou consumam muitos recursos computacionais. O atendimento desses requerimentos não seria uma preocupação atrelada ao dispositivo de captação e a sua limitação computacional.

No software principal também cabe fazer avaliações de desempenho, a partir do desenvolvimento de testes de carga que verifiquem a robustez do mesmo e do levantamento de alguns parâmetros, como: a latência entre envio do sinal por um software captador e o recebimento do mesmo pelo software principal; a latncia de envio de cada amostra do software principal até o banco de dados de série temporal, num caso de um volume alto de dados (a determinar); entre outros.

Para a aplicação web, é necessário finalizar a implementação da mesma, com base no planejamento descrito neste trabalho e no código desenvolvido até então. Além disso, é possível adicionar funcionalidades que correspondam à utilização das entidades novas acima citadas, permitindo o cadastro de clínicas, usuários e profissionais com relações entre si.

Ao software captador do sinal, já que o ESP32 tem suporte para Bluetooth, poderia ser dada a flexibilidade de configurar via tal tecnologia as credenciais da rede WiFi e o id_user de

quem está captando os sinais. Outra possibilidade é expandir seu uso para captar outros sinais vitais, a exemplo do ECG.

É possível também criar uma aplicação móvel que se comunique tanto com o software captador de sinal quanto com o software principal. No caso de uma comunicação com o software principal, essa aplicação poderia ter as mesmas funcionalidades da aplicação web, porém, com uma interface apropriada para a nova plataforma. Ademais, já que a aplicação móvel teria acesso ao id_user, ela poderia ser utilizada para configurar o software de captação de sinal para definir de qual id_user o sinal está sendo captado. Essas configurações poderiam ser feitas via Bluetooth, seguindo a incrementação do captador de sinal citada anteriormente.

Referências

AL-KABABJI, A. et al. IoT-based fall and ECG monitoring system: Wireless communication system based firebase realtime database. In: *2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. IEEE, 2019. Disponível em: <https://doi.org/10.1109/smartworld-uic-atc-scalcom-iop-sci.2019.00267>. Citado 2 vezes nas páginas 29 e 31.

ATHAVALE, Y.; KRISHNAN, S. Biosignal monitoring using wearables: Observations and opportunities. *Biomedical Signal Processing and Control*, Elsevier BV, v. 38, p. 22–33, set. 2017. Disponível em: <https://doi.org/10.1016/j.bspc.2017.03.011>. Citado 2 vezes nas páginas 28 e 31.

BADAWAY, G. A. J. W. *System-on-chip for real-time applications*. [S.l.]: Kluwer Academic Publishers, 2003. (The Kluwer international series in engineering and computer science). ISBN 9781402072543. Citado na página 21.

BENATTI, S. et al. A versatile embedded platform for EMG acquisition and gesture recognition. *IEEE Transactions on Biomedical Circuits and Systems*, Institute of Electrical and Electronics Engineers (IEEE), v. 9, n. 5, p. 620–630, out. 2015. Disponível em: <https://doi.org/10.1109/tbcas.2015.2476555>. Citado 2 vezes nas páginas 29 e 31.

BIAGETTI, G. et al. Wireless surface electromyograph and electrocardiograph system on 802.15.4. *IEEE Transactions on Consumer Electronics*, Institute of Electrical and Electronics Engineers (IEEE), v. 62, n. 3, p. 258–266, ago. 2016. Disponível em: <https://doi.org/10.1109/tce.2016.7613192>. Citado 2 vezes nas páginas 26 e 31.

BROWN, E. *Web development with Node and Express*. First edition. [S.l.]: O'Reilly, 2014. ISBN 9781491949306. Citado na página 21.

BRUNELLI, D. et al. Design considerations for wireless acquisition of multichannel sEMG signals in prosthetic hand control. *IEEE Sensors Journal*, Institute of Electrical and Electronics Engineers (IEEE), p. 1–1, 2016. Disponível em: <https://doi.org/10.1109/jsen.2016.2596712>. Citado 2 vezes nas páginas 29 e 31.

CAMERON, N. *Electronics projects with the ESP8266 and ESP32: building web pages, applications, and Wifi enabled devices*. Apress L. P., 2021. ISBN 9781484263365. Disponível em: <http://public.eblib.com/choice/PublicFullRecord.aspx?p=6427473>. Citado na página 22.

CANTELON, M. et al. *Node.js in action*. [S.l.]: Manning, 2014. ISBN 9781617290572. Citado na página 21.

CROCKFORD, D. *JavaScript: the good parts; [unearthing the excellence in JavaScript]*. 1. ed. ed. [S.l.]: O'Reilly, 2008. (Unearthing the excellence in JavaScript). ISBN 9780596517748. Citado na página 20.

DIX, P. *The Importance of Time Series Database | InfluxData Paper*. 2020. Disponível em: <https://www.influxdata.com/resources/

why-time-series-matters-for-metrics-real-time-and-sensor-data/>. Citado na página 17.

DUNNING, T. et al. *Time series databases: new ways to store and access data*. First edition. [S.l.]: O'Reilly Media, Inc, 2014. ISBN 9781491914724. Citado na página 17.

EDINGTON, D. W. et al. The future of health promotion in the 21st century: A focus on the working population. *American Journal of Lifestyle Medicine*, v. 10, n. 4, p. 242–252, Jul 2016. ISSN 1559-8276, 1559-8284. Citado na página 11.

ELMASRI, R.; NAVATHE, S. *Fundamentals of database systems*. Seventh edition. [S.l.]: Pearson, 2016. ISBN 9780133970777. Citado na página 15.

FARJADIAN, A. B.; SIVAK, M. L.; MAVROIDIS, C. SQUID: Sensorized shirt with smartphone interface for exercise monitoring and home rehabilitation. In: *2013 IEEE 13th International Conference on Rehabilitation Robotics (ICORR)*. IEEE, 2013. Disponível em: <https://doi.org/10.1109/icorr.2013.6650451>. Citado 2 vezes nas páginas 25 e 31.

FLYNN, M. J.; LUK, W. *Computer system design: system-on-chip*. [S.l.]: Wiley, 2011. ISBN 9780470643365. Citado na página 21.

GARCIA-MOLINA, H.; ULLMAN, J. D.; WIDOM, J. *Database systems: the complete book*. 2nd ed. ed. [S.l.]: Pearson Prentice Hall, 2009. ISBN 9780131873254. Citado na página 15.

GOURLEY, D.; TOTTY, B. *HTTP: the definitive guide*. 1st ed. ed. [S.l.]: O'Reilly, 2002. ISBN 9781565925090. Citado 2 vezes nas páginas 17 e 18.

HAKONEN, M.; PIITULAINEN, H.; VISALA, A. Current state of digital signal processing in myoelectric interfaces and related applications. *Biomedical Signal Processing and Control*, Elsevier BV, v. 18, p. 334–359, abr. 2015. Disponível em: <https://doi.org/10.1016/j.bspc.2015.02.009>. Citado 2 vezes nas páginas 27 e 31.

HAVERBEKE, M. *Eloquent javascript: a modern introduction to programming*. Second edition. [S.l.]: No Starch Press, 2015. ISBN 9781593275846. Citado na página 20.

HILLAR, G. C. *MQTT essentials: a lightweight IoT protocol: the preferred IoT publish-subscribe lightweight messaging protocol*. [S.l.]: Packt, 2017. ISBN 9781787287815. Citado na página 18.

KING, R. C. et al. Application of data fusion techniques and technologies for wearable health monitoring. *Medical Engineering & Physics*, Elsevier BV, v. 42, p. 1–12, abr. 2017. Disponível em: <https://doi.org/10.1016/j.medengphy.2016.12.011>. Citado 2 vezes nas páginas 27 e 31.

KITCHENHAM, B. A. *Procedures for Performing Systematic Reviews*. Department of Computer Science, Keele University, Kelee, UK, 2004. Disponível em: <http://www.it.hiof.no/~haraldh/misc/2016-08-22-smat/Kitchenham-Systematic-Review-2004.pdf>. Citado 2 vezes nas páginas 23 e 44.

LEE, B. G. et al. Wearable mobile-based emotional response-monitoring system for drivers. *IEEE Transactions on Human-Machine Systems*, Institute of Electrical and Electronics Engineers (IEEE), v. 47, n. 5, p. 636–649, out. 2017. Disponível em: <https://doi.org/10.1109/thms.2017.2658442>. Citado 2 vezes nas páginas 26 e 31.

LEE, S. M. et al. Design of an EMG signal recognition system for human-smartphone interface. In: *2015 International SoC Design Conference (ISOCC)*. IEEE, 2015. Disponível em: <https://doi.org/10.1109/isocc.2015.7401725>. Citado 2 vezes nas páginas 25 e 31.

LLORET, J. et al. An architecture and protocol for smart continuous eHealth monitoring using 5g. *Computer Networks*, Elsevier BV, v. 129, p. 340–351, dez. 2017. Disponível em: <https://doi.org/10.1016/j.comnet.2017.05.018>. Citado 2 vezes nas páginas 26 e 31.

LOMBARDI, A. *Websocket*. First edition. [S.l.]: O'Reilly, 2015. ISBN 9781449369279. Citado na página 18.

MASSé, M. H.; MASSé, M. *REST API design rulebook: designing consistent RESTful Web Service Interfaces*. [S.l.]: O'Reilly, 2012. ISBN 9781449310509. Citado 2 vezes nas páginas 19 e 20.

NAPOLI, M. L. *Beginning flutter: a hands on guide to app development*. [S.l.]: John Wiley and Sons, 2019. ISBN 9781119550822. Citado na página 21.

PANCHOLI, S.; JOSHI, A. M. Portable EMG data acquisition module for upper limb prosthesis application. *IEEE Sensors Journal*, Institute of Electrical and Electronics Engineers (IEEE), v. 18, n. 8, p. 3436–3443, abr. 2018. Disponível em: <https://doi.org/10.1109/jsen.2018.2809458>. Citado 2 vezes nas páginas 28 e 31.

PIND, R.; MäESTU, J. Monitoring training load: necessity, methods and applications. *Acta Kinesiologiae Universitatis Tartuensis*, v. 23, p. 7, Jan 2018. ISSN 2228-3501, 1406-9822. Citado na página 11.

PLEWA, K. et al. Designing a wearable MMG-based mobile app for gait rehab. In: *2017 IEEE Life Sciences Conference (LSC)*. IEEE, 2017. Disponível em: <https://doi.org/10.1109/lsc.2017.8268187>. Citado 2 vezes nas páginas 28 e 31.

PUDER, A.; ROMER, K.; PILHOFER, F. *Distributed Systems Architecture.* Elsevier Science, 2011. ISBN 9780080454702. Disponível em: <http://www.totalboox.com/book/id-7347845591668555345>. Citado na página 17.

PUENTE, S.; ÚBEDA, A.; TORRES, F. e-health: Biomedical instrumentation with arduino. *IFAC-PapersOnLine*, Elsevier BV, v. 50, n. 1, p. 9156–9161, jul. 2017. Disponível em: <https://doi.org/10.1016/j.ifacol.2017.08.1724>. Citado 2 vezes nas páginas 27 e 31.

PULVER, T. *Hands-on Internet of Things with MQTT: build connected IoT devices with Arduino and MQ Telemetry Transport (MQTT)*. First published. [S.l.]: Packt>, 2019. ISBN 9781789341782. Citado na página 19.

RAEZ, M.; HUSSAIN, M.; MOHD-YASIN, F. Techniques of emg signal analysis: detection, processing, classification and applications. *Biological Procedures Online*, v. 8, p. 11–35, Mar 2006. ISSN 1480-9222. Citado na página 22.

RICHARDSON, L.; AMUNDSEN, M. *RESTful Web APIs*. First edition. [S.l.]: O'Reilly, 2013. ISBN 9781449358068. Citado na página 20.

RICHER, R. et al. Real-time ECG and EMG analysis for biking using android-based mobile devices. In: *2014 11th International Conference on Wearable and Implantable Body Sensor Networks*. IEEE, 2014. Disponível em: <https://doi.org/10.1109/bsn.2014.20>. Citado 2 vezes nas páginas 26 e 31.

RIGHETTI, R. F. et al. Physiotherapy care of patients with coronavirus disease 2019 (covid-19) - a brazilian experience. *Clinics*, v. 75, 2020. ISSN 1807-5932, 1980-5322. Citado na página 12.

RODRIGUEZ, R. Y.; JULCAPOMA, M. R. Implementation of a sensor node for monitoring physiological signals with websocket communication and data visualization in a node.js server over the internet. In: *2020 IEEE Engineering International Research Conference (EIRCON)*. IEEE, 2020. Disponível em: <https://doi.org/10.1109/eircon51178.2020.9254049>. Citado 2 vezes nas páginas 29 e 31.

SADALAGE, P. J.; FOWLER, M. *NoSQL distilled: a brief guide to the emerging world of polyglot persistence*. [S.l.]: Addison-Wesley, 2013. ISBN 9780321826626. Citado na página 16.

SADIKOGLU, F.; KAVALCIOGLU, C.; DAGMAN, B. Electromyogram (emg) signal detection, classification of emg signals and diagnosis of neuropathy muscle disease. *Procedia Computer Science*, v. 120, p. 422–429, 2017. ISSN 18770509. Citado na página 22.

SARAEE, E. et al. Exercisecheck: Remote monitoring and evaluation platform for home based physical therapy. In: *Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments*. ACM, 2017. p. 87–90. ISBN 9781450352277. Disponível em: <https://dl.acm.org/doi/10.1145/3056540.3064958>. Citado na página 11.

SHAOWN, T. et al. IoT-based portable ECG monitoring system for smart healthcare. In: *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*. IEEE, 2019. Disponível em: <https://doi.org/10.1109/icasert.2019.8934622>. Citado 2 vezes nas páginas 30 e 31.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. *Database system concepts*. Seventh edition. [S.l.]: McGraw-Hill, 2020. ISBN 9780078022159. Citado 2 vezes nas páginas 15 e 16.

TEDESCO, S. et al. A multi-sensors wearable system for remote assessment of physiotherapy exercises during acl rehabilitation. In: *2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*. IEEE, 2019. p. 237–240. ISBN 9781728109961. Disponível em: <https://ieeexplore.ieee.org/document/8965214/>. Citado na página 12.

TIWARI, S. *Professional NoSQL*. [S.l.]: Wiley, 2011. (Wrox programmer to programmer). ISBN 9780470942246. Citado na página 16.

TROST, S. G.; BLAIR, S. N.; KHAN, K. M. Physical inactivity remains the greatest public health problem of the 21st century: evidence, improved methods and solutions using the '7 investments that work' as a framework. *British Journal of Sports Medicine*, v. 48, n. 3, p. 169–170, Feb 2014. ISSN 0306-3674, 1473-0480. Citado na página 11.

WANG, H.-P. et al. A wearable multi-pad electrode prototype for selective functional electrical stimulation of upper extremities. In: *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2017. Disponível em: <https://doi.org/10.1109/embc.2017.8036924>. Citado 2 vezes nas páginas 28 e 31.

WETMORE, A. Sport science on a budget: A cost effective approach for athlete monitoring. In: . [S.l.: s.n.], 2016. Citado na página 12.

WINDMILL, E.; RISCHPATER, R. *Flutter in action*. [S.l.]: Manning Publications Co, 2020. ISBN 9781617296147. Citado na página 21.

YURTMAN, A.; BARSHAN, B. Automated evaluation of physical therapy exercises using multi-template dynamic time warping on wearable sensor signals. *Computer Methods and Programs in Biomedicine*, Elsevier BV, v. 117, n. 2, p. 189–207, nov. 2014. Disponível em: <https://doi.org/10.1016/j.cmpb.2014.07.003>. Citado 2 vezes nas páginas 27 e 31.