UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE CIÊNCIAS

DEPARTAMENTO DE COMPUTAÇÃO

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

NADIANA KELLY NOGUEIRA MENDES

INTEGRAÇÃO DE SISTEMAS DE SAÚDE DIGITAL PARA DETECÇÃO DE QUEDAS EM SISTEMAS HOSPITALARES

FORTALEZA

2024

NADIANA KELLY NOGUEIRA MENDES

INTEGRAÇÃO DE SISTEMAS DE SAÚDE DIGITAL PARA DETECÇÃO DE QUEDAS EM SISTEMAS HOSPITALARES

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Orientadora: Prof. Dra. Rossana Maria de Castro Andrade.
Coorientador: Me. Francisco Victor da Silva Pinheiro.

FORTALEZA

2024

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

M492i Mendes, Nadiana Kelly Nogueira.
Integração de sistemas de saúde digital para detecção de quedas em sistemas hospitalares / Nadiana Kelly Nogueira Mendes. – 2024.
61 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Centro de Ciências, Curso de Computação, Fortaleza, 2024.
Orientação: Profa. Dra. Rossana Maria de Castro Andrade.
Coorientação: Prof. Me. Francisco Victor da Silva Pinheiro.

1. Integração de sistemas. 2. Saúde digital. 3. Internet of health things. 4. Application programming interface. I. Título.

CDD 005

NADIANA KELLY NOGUEIRA MENDES

INTEGRAÇÃO DE SISTEMAS DE SAÚDE DIGITAL PARA DETECÇÃO DE QUEDAS EM SISTEMAS HOSPITALARES

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Aprovada em:

BANCA EXAMINADORA

Prof. Dra. Rossana Maria de Castro Andrade (Orientadora)
Universidade Federal do Ceará (UFC)

Me. Francisco Victor da Silva Pinheiro (Coorientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Rafael Lopes Gomes
Universidade Estadual do Ceará (UECE)

Prof. Dr. Ítalo Linhares de Araújo
Universidade Federal do Ceará (UFC)

À minha mãe, por sempre acreditar em mim e investir em meu futuro. Mãe, foi o seu carinho e sua dedicação que, em muitos momentos, me renovaram a esperança para continuar. Sua presença trouxe a segurança e a confiança de que eu nunca estive sozinha nesta jornada.

AGRADECIMENTOS

Agradeço a Deus por me conceder força, sabedoria e perseverança ao longo de todo o meu percurso acadêmico.

Agradeço à minha mãe pelo apoio incondicional e pela motivação durante toda a minha jornada acadêmica. Seu constante encorajamento e crença em meu potencial foram fundamentais para que eu chegasse até aqui.

Quero dedicar um agradecimento especial ao meu namorado, Wanderley, a quem tive a sorte de conhecer durante essa jornada acadêmica e que tem sido meu apoio desde então. Sua presença constante e seu carinho foram essenciais para mim. Nos momentos mais difíceis, ele esteve ao meu lado, oferecendo apoio e força. Sou imensamente grata por tê-lo, não apenas como parceiro, mas como alguém que compartilha e acredita nos meus sonhos.

Meus agradecimentos aos amigos que fiz durante essa jornada, em especial ao meu amigo Silvio, por estar ao meu lado em todos os momentos. Sua presença nas alegrias e dificuldades e seu apoio foram essenciais para superar os desafios. Sua amizade foi um pilar ao longo de todo o percurso, e espero que possamos manter esse vínculo especial por toda a vida. Agradeço também ao meu amigo Matheus do Vale pela amizade e apoio ao longo dessa jornada. Além disso, agradeço a todos os amigos que fiz na faculdade por estarem ao meu lado e compartilharem momentos inesquecíveis.

Gostaria de expressar minha mais profunda gratidão à Prof. Dra. Rossana, que me deu todo apoio ao longo do desenvolvimento deste projeto. Agradeço profundamente pela sua dedicação e pelo tempo investido em orientar este trabalho. Obrigada por me proporcionar um aprendizado que vai além deste TCC, contribuindo para a minha formação acadêmica e profissional.

Agradeço ao Victor Pinheiro pelo apoio e pelas contribuições oferecidas durante a elaboração deste trabalho. Obrigada por suas orientações, que complementaram e fortaleceram este trabalho.

Agradeço à Universidade Federal do Ceará por me proporcionar uma educação de qualidade e por todas as oportunidades que contribuíram para a realização deste trabalho e para o meu crescimento acadêmico. Por fim, agradeço ao Programa Institucional de Bolsas de Iniciação Científica (PIBIC) do CNPq.

"Todos os dias de nossas vidas estamos prestes a fazer aquelas pequenas mudanças que fariam toda a diferença." (Mignon McLaughlin)

RESUMO

Quedas em idosos podem levar a ferimentos sérios e até fatalidades, considerando a fragilidade dessa faixa etária. Assim, é importante o desenvolvimento de sistemas que auxiliem no monitoramento e atendimento de idosos em casos de queda. Na saúde, a Internet das Coisas (IoT) tem crescido, com aplicações que oferecem desde cuidados emergenciais até intervenções a longo prazo, impulsionando a criação de novos sistemas. Um exemplo é o sistema *Health Risk*, que detecta quedas em pacientes, utilizando dispositivos vestíveis e aprendizado de máquina, com uma precisão de 94,4%. Este trabalho descreve o desenvolvimento de uma API para a integração de dois sistemas de saúde: um sistema mobile de detecção de quedas baseado em Internet das Coisas, o *Health Risk*, e um sistema hospitalar projetado para um hospital universitário, o *Smart Hospital*. Para a integração desses sistemas, foi feita uma revisão da literatura sobre diversos temas relacionados à Internet das Coisas com o intuito de realizar um estudo aprofundado sobre os padrões no processo de desenvolvimento e exemplos de aplicações bem-sucedidas envolvendo sistemas baseados no paradigma de Internet das Coisas integrados a outros sistemas de saúde. Em seguida, a API proposta neste trabalho, denominada de *FallReportAPI*, responsável pela integração do *Health Risk* e *Smart Hospital*, foi construída utilizando Java como linguagem de programação, *Spring Boot* para a criação das rotas, *RabbitMQ* para a troca de dados e *PostgreSQL* como base de dados. Para avaliar a integração completa dos sistemas, foram realizados testes de API focando na realização de testes funcionais para garantir a obtenção dos resultados esperados. Com a integração desses sistemas feita pela API desenvolvida neste trabalho, a comunicação entre os sistemas resulta em atualizações imediatas em caso de queda dos usuários do aplicativo *Health Risk* para o sistema do *Smart Hospital*. Com essa comunicação ágil entre os sistemas de saúde, espera-se reduzir o atendimento para o chamado da equipe de saúde em caso de quedas que necessitem de atendimento médico urgente.

Palavras-chave: Integração de Sistemas. Saúde Digital. Internet of Health Things (IoHT). Application Programming Interface (API).

ABSTRACT

Falls in the elderly can lead to serious injuries and even fatalities, considering the vulnerability of this age group. Thus, it is important to develop systems that assist in monitoring and caring for elderly individuals in cases of falls. In healthcare, the Internet of Things (IoT) has been growing, with applications that offer everything from emergency care to long-term interventions, driving the creation of new systems. An example is the Health Risk system, which detects falls in patients using wearable devices and machine learning, with an accuracy of 94.4%. This work describes the development of an API for the integration of two healthcare systems: a mobile fall detection system based on the Internet of Things, Health Risk, and a hospital system designed for a university hospital, the Smart Hospital. To integrate these systems, a literature review was conducted on various topics related to the Internet of Things to carry out an in-depth study of standards in the development process and examples of successful applications involving IoT-based systems integrated with other healthcare systems. Subsequently, the API proposed in this work, named FallReportAPI, which is responsible for the integration of Health Risk and Smart Hospital, was built using Java as the programming language, Spring Boot for creating the routes, RabbitMQ for data exchange, and PostgreSQL as the database. To evaluate the complete integration of the systems, API tests were conducted focusing on functional tests to ensure the expected results were achieved. With the integration of these systems facilitated by the API developed in this work, communication between the systems results in immediate updates in case of falls by users of the Health Risk app to the Smart Hospital system. With this agile communication between healthcare systems, it is expected to reduce response time for healthcare teams in case of falls that require urgent medical attention.

Keywords: Systems Integration. Healthcare. Internet of Health Things (IoHT). Application Programming Interface (API).

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
BIM	*Building Information Modeling*
CNN	*Convolutional Neural Network*
CPF	Cadastro de Pessoas Físicas
GREat	Grupo de Redes de Computadores, Engenharia de Software e Sistemas
IoHT	*Internet of Health Things*
IoMT	*Internet of Medical Things*
IoT	*Internet of Things*
JSON	*JavaScript Object Notation*
MBAN	*Mobile Body Area Network*
MQTT	*Message Queuing Telemetry Transport*
OIS	Objetos Inteligentes de Saúde
SAMU	Serviço de Atendimento Móvel de Urgência
SUS	Sistema Único de Saúde
TARM	Técnicos de Atendimento e Regulação Médica

SUMÁRIO

1 INTRODUÇÃO

Este Capítulo está dividido da seguinte maneira: na Seção 1.1 é apresentado o contexto em que este trabalho está inserido; na Seção 1.2 são apresentados a motivação e o objetivo desta monografia; na Seção 1.3 são apresentadas as questões de pesquisa deste trabalho; na Seção 1.4 é apresentada a metodologia que baseia o desenvolvimento do trabalho apresentado; por fim, na Seção 1.5 é apresentada a estrutura do trabalho como um todo.

1.1 Contextualização

A Internet das Coisas também conhecida como *Internet of Things* (IoT), oferece a qualquer objeto que tenha capacidade computacional e de comunicação de se conectar com outros objetos utilizando a internet, desde que esses objetos possuam a mesma condição (SANTOS *et al.*, 2016). Essa capacidade para objetos comuns oferece diversas possibilidades tanto no âmbito acadêmico quanto no industrial (SANTOS *et al.*, 2016). Com o uso crescente da tecnologia na sociedade atual é possível observar a frequência com a qual a tecnologia pode ajudar na solução de problemas da sociedade.

A saúde, sendo um dos principais pilares da sociedade, pode ser altamente beneficiada com o uso da tecnologia. Sendo assim, é crescente o uso de aplicações de Internet das Coisas voltadas para a área da saúde que possuem o potencial de oferecer cuidados aos pacientes em diversas configurações, podendo oferecer desde o tratamento de emergência no hospital até intervenções de longo prazo. Nesse contexto, a Internet das Coisas da saúde se destaca significativamente na área da tecnologia, impulsionando cada vez mais a criação de novos sistemas que colaboram com os avanços na saúde (SILVA; OLIVEIRA, 2017).

Em meio a este cenário foi tomada a iniciativa em projetos do Grupo de Redes de Computadores, Engenharia de Software e Sistemas (GREat) de criar um sistema de detecção de quedas, o *Health Risk*, fundamentado na ideia de Internet das Coisas. O sistema de detecção de quedas pode desempenhar um papel significativo na área da saúde, principalmente no tratamento de emergências envolvendo pacientes e idosos que necessitam de monitoramento. O sistema auxilia no socorro de pacientes em casos de quedas. Uma maneira de identificar quedas potenciais é utilizando dados coletados por dispositivos vestíveis como entrada para um sistema IoT, que emprega modelos de *Machine Learning* com suporte da computação em nuvem. O modelo utilizado no *Health Risk* atinge uma precisão de 94,4%, conseguindo distinguir corretamente

entre uma queda e um evento que não seja uma queda (ARAÚJO *et al.*, 2022). Para o uso do aplicativo é necessário um *smartphone Android* e um *Smartwatch* que possa interagir com a aplicação.

O *Smart Hospital* é um sistema hospitalar projetado para atender ao hospital universitário da Universidade Estadual do Ceará. O sistema foi desenvolvido como uma solução para diversos problemas enfrentados nas unidades de saúde do estado e representa uma inovação tecnológica em comparação com os *softwares* hospitalares já existentes e utilizados. Este projeto é fruto de uma parceria entre a Universidade Estadual do Ceará e a indústria Huawei (COSTA *et al.*, 2023).

1.2 Motivação e Objetivo

Com o propósito de aprimorar a integração do sistema de detecção de quedas com outras tecnologias, este trabalho visa o desenvolvimento de uma *Application Programming Interface* (API). Uma API é uma forma de comunicação entre sistemas (JUNIOR *et al.*, 2021). Ela facilita a integração entre dois sistemas, permitindo que um forneça informações e serviços para o outro, sem que o sistema que utiliza a API precise entender os detalhes da implementação do *software* fornecido (JUNIOR *et al.*, 2021). O objetivo geral deste trabalho é criar uma API, denominada *FallReportAPI*, que integra o sistema de detecção de quedas, *Health Risk*, com o sistema hospitalar *Smart Hospital*, visando contribuir para o aspecto hospitalar do atendimento de saúde. O objetivo específico deste trabalho é contribuir positivamente para a redução do tempo de atendimento nas unidades de saúde que utilizam o *Smart Hospital*, em situações de queda, com foco prioritário na população idosa.

Este trabalho propõe-se então a resolver um desafio na área da saúde, mais especificamente no contexto de monitoramento e atendimento de emergência relacionada a quedas. Esse desafio é o atendimento ágil das unidades de saúde em circunstâncias de quedas graves, com uma atenção voltada principalmente para idosos.

Para a integração desses dois sistemas, o *Health Risk* e o *Smart Hospital*, foi realizada primeiramente uma revisão extensiva da literatura para o estudo e a análise de diversos artigos relacionados ao uso de Internet das Coisas voltada para a saúde. Após esta fase, foi iniciado um estudo no código fonte do *Health Risk*. Inicialmente, foi realizada uma análise da aplicação sob a perspectiva do usuário. Posteriormente, foi feita uma análise detalhada do código da aplicação. Depois dessa análise, foi realizado um planejamento focado nas necessidades da construção

da API. Por último, foi feita a escolha das tecnologias de desenvolvimento para a codificação desta API, que se chama *FallReportAPI*. Optou-se por utilizar o *PostgreSQL* como banco de dados, *RabbitMQ* para a troca de mensagens de dados, Java como linguagem de programação e o *framework Spring Boot* para a construção das rotas. Ao concluir essa etapa ocorreu a fase de teste de API que assegura a comunicação entre as duas aplicações.

Ao conduzir o desenvolvimento da *FallReportAPI* e analisar os resultados obtidos com a integração desses dois sistemas, este trabalho pretende auxiliar na criação de outras APIs que integrem sistemas de saúde com sistemas de Internet das Coisas da saúde e promovam adicionalmente mais agilidade no atendimento de saúde. Além disso, visa-se a contribuição nos estudos relacionados ao uso de tecnologias baseadas em *Internet of Health Things* (IoHT) e seus resultados promissores em ambientes hospitalares.

1.3 Questões de Pesquisa

Dentro deste escopo, as questões de pesquisa que guiam este trabalho estão detalhadas a seguir:

(Q1) Como um sistema mobile de detecção de quedas baseado em Internet das Coisas e um sistema hospitalar podem ser integrados colaborando com uma comunicação entre si?

(Q2) Qual é o impacto da integração de um aplicativo de detecção de quedas com um sistema hospitalar no atendimento de emergência relacionada a quedas?

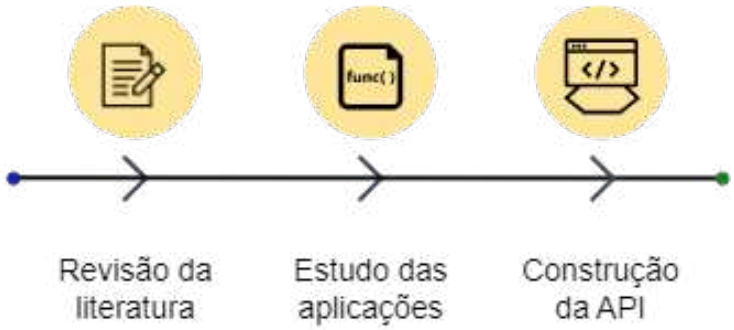
Para responder a **(Q1)** é considerado o desenvolvimento da API deste trabalho focando na colaboração entre os sistemas que serão integrados e nas tecnologias que integram essas aplicações como um todo.

Para a **(Q2)** é considerada a definição da *International Organization for Standardization*, especificamente a ISO/IEC 25010 (ISO, 2011) no que se refere à definição de característica de qualidade de *software* focada na eficiência de um produto. A ISO cria normas internacionais para assegurar a qualidade, segurança e confiabilidade de produtos e serviços. A série de normas ISO/IEC 25000 (ISO, 2014), também conhecida como *SQuaRE*, *System and Software Quality Requirements and Assessmen*, aborda vários aspectos da qualidade de software, incluindo o comportamento temporal. A ISO/IEC 25010 define eficiência de desempenho como o desempenho em comparação com a quantidade de recursos empregados sujeito às condições determinadas. Com essa definição, é possível estudar o tempo de resposta de um atendimento após uma queda,

comparando o tempo de resposta de um atendimento utilizando a *FallReportAPI*, com o tempo de resposta de um atendimento não fazendo uso da API.

1.4 Metodologia

Figura 1 – Metodologia



Fonte: Elaborado pelo autor.

Conforme apresentado na Figura 1, a metodologia é composta pelos seguintes passos: revisão da literatura, que consiste em uma busca e análise de trabalhos relacionados; estudo das aplicações, no qual o código das aplicações envolvidas, que são os sistemas *Health Risk* e *Smart Hospital*, é investigado e compreendido; e construção da API, que consiste na implementação da integração dos sistemas e avaliação da mesma.

1.4.1 Revisão da Literatura

Antes de iniciar o desenvolvimento da API, é necessário realizar uma revisão da literatura sobre os diversos temas relacionados à IoT no contexto da saúde. Nesta etapa, é conduzido um estudo aprofundado de artigos e publicações pertinentes ao tema da IoHT. Com essa revisão, é possível compreender a diversidade de aplicações que podem ser desenvolvidas para auxiliar a saúde utilizando tecnologia baseada na ideia de Internet das Coisas. A análise da literatura permite absorver ideias relacionadas ao tema, padrões no processo e desenvolvimento, e exemplos de aplicações desenvolvidas com sucesso. Além de entender o funcionamento de cada sistema, esse processo de estudo fortalece o desenvolvimento da API proposta, visando ampliar sua relevância na área de saúde.

1.4.2 Estudo das Aplicações

Após a revisão da literatura, é necessário realizar um estudo das aplicações envolvidas no desenvolvimento do trabalho apresentado, para que sejam analisadas as conexões necessárias entre os sistemas pela integração. Em seguida, é preciso coletar os requisitos necessários para o desenvolvimento da API, com o intuito de desenvolver corretamente as estruturas necessárias para a integração dos sistemas.

1.4.3 Construção da API

Com a etapa de estudo das aplicações finalizada, é necessário avançar para a etapa de implementação da API propriamente dita, que constitui o desenvolvimento principal deste trabalho. Isso deve ser feito utilizando as tecnologias adequadas e os requisitos definidos de forma objetiva para sua execução. Por fim, ainda dentro desta etapa, é fundamental o teste de API para comprovar a integração implementada entre os dois sistemas, *Smart Hospital* e *Health Risk*. O teste de API consiste em avaliar se a API está operando conforme o esperado, verificando se as funcionalidades estão corretas. Essa avaliação é realizada por meio do envio de solicitações a diferentes pontos de acesso da API, que são os *endpoints*, e da comparação das respostas obtidas com os resultados esperados.

1.5 Estrutura do Trabalho

Este trabalho está dividido em seis capítulos. Além deste capítulo introdutório, o Capítulo 2 apresenta a fundamentação teórica essencial para embasar o trabalho proposto; o Capítulo 3 exibe os trabalhos relacionados a este trabalho; o Capítulo 4 descreve detalhadamente o funcionamento da *FallReportAPI*; o Capítulo 5 mostra a validação dos resultados obtidos neste trabalho com o teste de integração e a discussão dos resultados alcançados; e o Capítulo 6 foca nos resultados obtidos neste trabalho e sugere trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta uma base de conhecimento teórico que sustenta a criação deste trabalho. Inicialmente será visto na Seção 2.1 e 2.2 os principais conceitos que formam a base deste trabalho, seguindo da Seção 2.3 que apresenta os estudos sobre o tema de quedas em população idosa, continuando com a Seção 2.4 que apresenta a análise sobre as principais tecnologias utilizadas para monitoramento de quedas em idosos e finalizando com a Seção 2.5 que destaca os principais sistemas envolvidos para a construção deste trabalho.

2.1 Internet das Coisas

Weiser escreveu o artigo *"The Computer for the 21st Century"*, que discute o futuro do uso das tecnologias, o que nos dias atuais se assemelha ao paradigma de Internet das Coisas. Ele afirmou que os dispositivos serão capazes de se conectar em todos os lugares de forma transparente para o ser humano de tal forma que se torne invisível possibilitando que a realização de atividades seja feita de forma natural sem preocupação de instalação, configuração e manter recursos computacionais (WEISER, 1999).

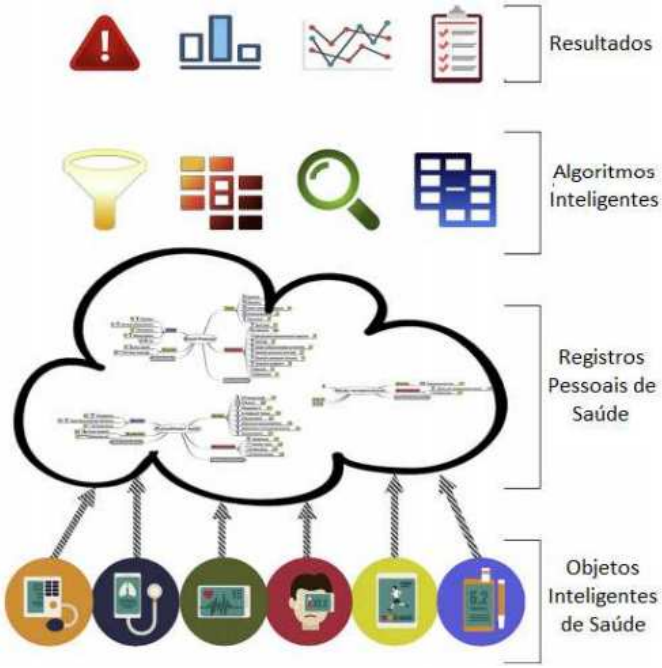
A Internet das Coisas, de forma simplificada, é uma extensão da internet atual que possibilita que objetos do nosso cotidiano, que possuem capacidade para computação e comunicação, se conectem à rede mundial. Essa ligação à internet nos permite, em primeiro lugar, controlar esses objetos de forma remota e, em segundo lugar, utilizar esses objetos como prestadores de serviços. Essas novas habilidades dos objetos que usamos todos os dias abrem diversas oportunidades tanto no meio acadêmico quanto na indústria (SANTOS *et al.*, 2016).

Existem várias aplicações que podem ser feitas com Internet das Coisas, atualmente, muitas mencionam a coleta de dados em várias configurações, capacidade de interagir diretamente com objetos de todas as naturezas, a conexão em rede e a comunicação entre os próprios objetos, bem como a interação entre objetos e pessoas, seja de maneira intencional ou discreta (FILHO, 2016). Analisando a literatura, é possível perceber a ampla gama de aplicações baseado na ideia de Internet das Coisas na área da saúde, que traz inovações para simplificar procedimentos, economizar tempo e recursos, minimizar erros causados pela intervenção humana, adaptar tratamentos de forma personalizada e fornecer dados para prevenir e gerenciar doenças (ROSA *et al.*, 2020).

2.2 Internet das Coisas da Saúde

A IoHT inclui todas as soluções de saúde baseadas na Internet das Coisas. Isso engloba sistemas hospitalares que utilizam IoT, sistemas de monitoramento de sinais vitais, como nível de oxigênio no sangue e pulso, utilizando dispositivos e sensores de IoT, e sistemas que aproveitam dados de vários sensores, como os de movimentação, para acompanhar a saúde e melhorar a qualidade de vida dos pacientes (RODRIGUES *et al.*, 2018). De acordo com (COSTA *et al.*, 2018), a IoHT envolve objetos interconectados que podem trocar e processar dados com o objetivo de melhorar a saúde dos pacientes. Essa abordagem centrada no paciente abrange quatro camadas diferentes, conforme mostradas na Figura 2.

Figura 2 – Visão geral de IoHT com quatro camadas



Fonte: (JUNIOR, 2023)

Em resumo, a camada de aquisição é responsável por coletar dados de sinais vitais ou de outras condições fisiológicas através de Objetos Inteligentes de Saúde (OIS). A camada de armazenamento, por sua vez, é responsável por esses dados de maneira escalável e interoperável, utilizando frequentemente computação em nuvem ou em névoa, além de prontuários eletrônicos. A camada de processamento analisa os dados dos pacientes com o auxílio de algoritmos de aprendizado de máquina, visando otimizar recursos. Por fim, a camada de apresentação visualiza os resultados das etapas anteriores, podendo assumir a forma de alertas, gráficos e tabelas.

É possível afirmar que com o rápido avanço da tecnologia juntamente com o promis-

sor desenvolvimento de sensores e atuadores no desenvolvimento de IoT, proporcionam grandes oportunidades para o desenvolvimento de sistemas inteligentes na área da saúde (YUEHONG *et al.*, 2016).

Com um sistema inteligente e algoritmos poderosos, é viável obter um volume sem igual de dados críticos em tempo real, os quais são capturados e analisados para orientar pesquisas, gestão e cuidados intensivos de forma avançada. O motivo para a integração dos cuidados de saúde com as funcionalidades da Internet das Coisas nos dispositivos médicos é aprimorar a qualidade e a eficiência do serviço, oferecendo um valor particularmente significativo para idosos, pacientes com doenças crônicas e aqueles que necessitam de monitoramento constante (BHATT *et al.*, 2017).

2.3 Estudos sobre Quedas de Idosos

O processo de envelhecimento da população é um desafio sem precedentes que praticamente todos os países estão enfrentando (CHEN *et al.*, 2019). Em 2019, cerca de 703 milhões de pessoas em todo o mundo tinham 65 anos ou mais, equivalente a 9% da população global. É esperado que esse número de idosos duplique, chegando a 1,5 bilhão até 2050, o que representará aproximadamente 16% da população mundial (NATIONS, 2015).

Dados revelaram que idosos vítimas de queda têm maiores chances para classificações em cores que sinalizam para um maior grau de prioridade, que são as cores laranja e vermelho o que significa muito urgente. As quedas, mesmo aquelas que ocorrem a partir da própria altura, podem resultar em lesões graves e potencialmente fatais (MARTINS *et al.*, 2022).

Alguns projetos que auxiliam na saúde dos idosos já existem, como, por exemplo, o desenvolvimento de uma plataforma voltada para melhorar a qualidade de vida dos idosos, integrando soluções de saúde e tecnologia (ALMEIDA *et al.*, 2019). Outro projeto como o desenvolvimento de aplicativo para auxiliar o profissional de saúde na administração da saúde de idosos que necessitam de cuidados especiais (BARRETO *et al.*, 2020). Já existe taxonomia criada com o objetivo de dar suporte ao desenvolvimento de aplicativos de software voltados para a saúde dos idosos, ou seja, um sistema de classificação que organiza as características e informações essenciais para o desenvolvimento de aplicativos de software que estão ligados à saúde dos idosos.(ARAÚJO *et al.*, 2020)

2.4 Tecnologias para a Saúde

Estudos revelam que os *Smartphones* são os dispositivos mais comuns em uso na área da saúde. Por exemplo, o uso de *Smartphones* na prevenção de quedas se apresenta como uma solução de baixo custo, tanto para cuidadores e familiares quanto para o sistema de saúde em si. A maioria desses dispositivos já vem equipada ou pode receber aplicativos com funcionalidades como bússola, GPS, microfone, câmera, acelerômetros e giroscópio, o que amplia e aprimora a capacidade de monitorar a mobilidade de idosos ao longo do dia (NASCIMENTO *et al.*, 2022).

Existe um sistema que já utiliza API para integrar sistemas no setor da saúde, atendendo a diversas necessidades, como, por exemplo, fornecer uma visão unificada e compartilhada dos dados essenciais de um indivíduo para qualquer provedor de cuidados de saúde envolvido em seu círculo de cuidados, independentemente da localização física do provedor ou do indivíduo, ou da organização à qual pertencem (AZARM *et al.*, 2017).

A pesquisa sobre tecnologias voltadas para a prevenção de quedas no ambiente hospitalar traz valiosas contribuições para a gestão administrativa em colaboração com profissionais de saúde. Isso permite identificar e adquirir os recursos mais adequados ao contexto econômico e social, considerando os indicadores de assistência e o perfil epidemiológico da população idosa atendida. É importante ressaltar que a aquisição desses dispositivos requer um investimento inicial. No entanto, os resultados podem oferecer uma redução de custos para o sistema de saúde, decorrente da diminuição do tempo de internação e da necessidade de tratamentos adicionais causados por quedas. Além disso, essas tecnologias proporcionam mais segurança para a equipe de saúde e uma melhor qualidade de vida para os idosos atendidos (ALVES *et al.*, 2022).

2.5 Sistemas de Saúde Digital

Esta monografia utiliza dois sistemas que constituem os pilares principais do desenvolvimento deste trabalho, um deles é o *Health Risk*, um sistema mobile de detecção de quedas e o outro sistema é o *Smart Hospital*. Nesta seção, os dois sistemas são descritos.

2.5.1 Health Risk: Sistema de Detecção de Quedas

O *Health Risk* teve início como um dos projetos do Grupo de Redes de Computadores, Engenharia de Software e Sistemas (GREat) [1]. A aplicação é um sistema mobile para

[1] https://www.great.ufc.br/

detecção de quedas, ele executa em um *Smartwatch* e um *Smartphone*. O *Health Risk* foi desenvolvido para atender à crescente necessidade de monitorar idosos que precisam de cuidados especiais, especialmente aqueles que passam a maior parte do tempo sozinhos. O sistema monitora em tempo real a movimentação do usuário, detectando quedas por meio da análise dos dados dos sensores e realiza a emissão de alertas de ajuda quando necessário. O aplicativo foi desenvolvido para a plataforma *Android* e faz uso dos seguintes sensores: acelerômetro e giroscópio. (ANDRADE *et al.*, 2021; LINHARES *et al.*, 2020).

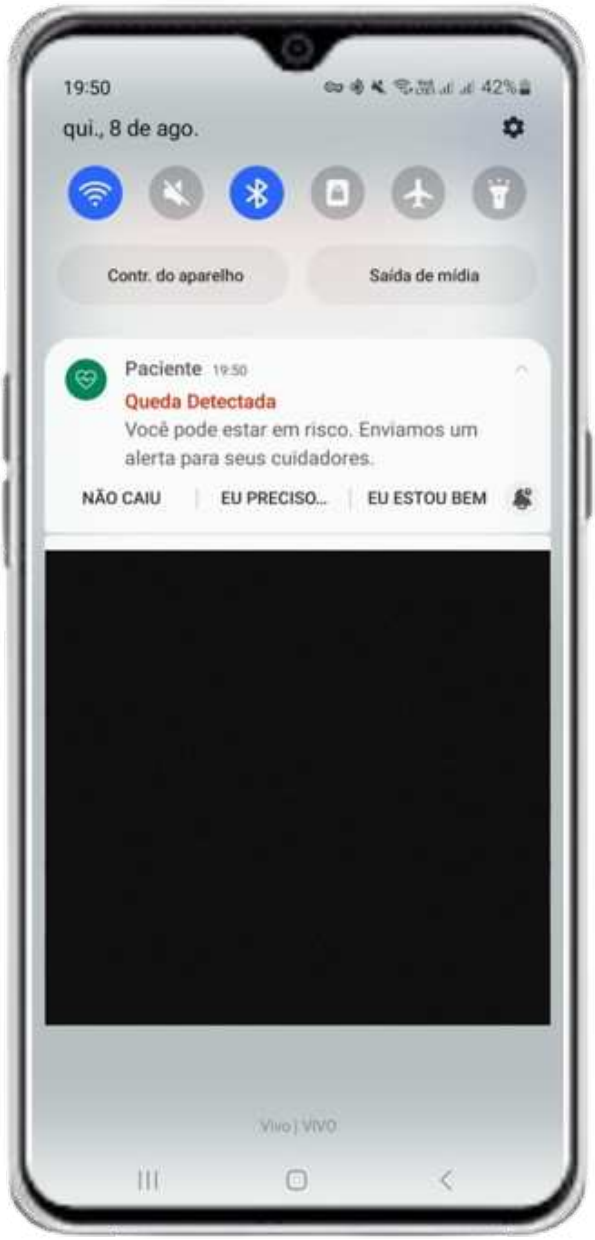
Figura 3 – Tela inicial do aplicativo Health Risk

Fonte: Elaborado pelo autor.

Quando uma queda é detectada e não há resposta ou confirmação de bem-estar, o

Smartwatch se conecta ao *Smartphone*, que então envia um alerta de queda para o contato de um cuidador, responsável pelo usuário, previamente registrado. Para detectar possíveis quedas, são utilizados dados coletados de dispositivos vestíveis como entrada para um sistema IoT. Esse sistema emprega modelos de aprendizado de máquina e, no caso do *Health Risk*, também faz uso da computação em nuvem. O algoritmo utilizado no *Health Risk* para a detecção de quedas apresenta uma precisão de 94,4%, com uma baixa taxa de falsos negativos de 4,3% (ARAÚJO *et al.*, 2022). Imagens do aplicativo *Health Risk* podem ser vistas nas Figura 3 e 4.

Figura 4 – Notificação de possível queda do aplicativo Health Risk



Fonte: Elaborado pelo autor.

2.5.2 *Smart Hospital: Sistema de Gerenciamento Hospitalar*

Outro sistema importante para a construção deste trabalho é o *Smart Hospital*, que foi desenvolvido pela Universidade Estadual do Ceará (UECE) em parceria com o Grupo de Redes de Computadores, Engenharia de Software e Sistemas (GREat) da Universidade Federal do Ceará (UFC), tendo como parceiro da indústria, a Huawei [2] (COSTA *et al.*, 2023). O *Smart Hospital* surgiu com a necessidade criar um sistema de gerenciamento hospitalar para o hospital universitário da UECE. O *Smart Hospital* foi projetado para auxiliar o cotidiano dos profissionais de saúde de forma mais precisa e customizada nos desafios relacionados ao atendimento de pacientes, à gestão de pessoas e ao grande volume de usuários.

Figura 5 – Tela do sistema *Smart Hospital*

Fonte: Captura de tela do sistema *Smart Hospital*, elaborada pelo autor.

A aplicação representada pela Figura 5 possui muitos diferenciais, como a sua customização acerca da realidade de cada hospital, a opção de gestão múltipla, referente ao sistema gerenciar vários hospitais que utilizam o mesmo *software*. Além disso, o *Smart Hospital* é um sistema escalável e multiplataforma, a aplicação consegue lidar com um número crescente de usuários e possui um funcionamento adaptado para diferentes aparelhos e sistemas operacionais. Para a implementação desse sistema foram utilizadas as linguagens *Python* e *JavaScript*, e também as bibliotecas do *React*, *React Bootstrap*.

[2] https://www.huawei.com/br/

3 TRABALHOS RELACIONADOS

Neste capítulo serão apresentados artigos que apontam aspectos relacionados ao trabalho apresentado nesta monografia. A Seção 3.1 detalha como foi feita a busca na literatura pelos trabalhos relacionados e as Seções 3.2, 3.3, 3.4 e 3.5 tratam dos trabalhos propriamente ditos, que possuem o aspecto da integração de sistemas voltados para a área da saúde. Na Seção 3.6 deste capítulo, é apresentada uma tabela comparativa entre o trabalho desta monografia e os trabalhos relacionados apresentados neste capítulo.

3.1 Busca na Literatura

Para a busca de trabalhos relacionados, utilizou-se a plataforma *Google Scholar*. A escolha do *Google Scholar* foi motivada pela vasta quantidade de fontes de trabalhos acadêmicos disponíveis. A plataforma é fácil de usar, oferece uma variedade de resultados e facilita o acesso a versões gratuitas de trabalhos, o que contribui significativamente para a busca de trabalhos relacionados. A busca foi realizada com as seguintes frases: *"Integration between healthcare systems and IoT systems"*, *"Fall detection and rescue"*, *"IoT systems for healthcare"*, visando identificar trabalhos que fossem especificamente voltados para a aplicação da Internet das Coisas na área da saúde. Essas expressões foram escolhidas para abranger tanto a integração de tecnologias IoT com sistemas de saúde quanto a detecção de quedas e os procedimentos de resgate, que são tópicos diretamente relevantes para o desenvolvimento do projeto. É fundamental destacar que a pesquisa realizada na plataforma *Google Scholar* também teve concentração em termos relacionados ao uso de APIs em aplicações para a saúde, como *"API use in healthcare systems"* e *"use of APIs for integration of IOHT systems"*. No entanto, com essas expressões específicas, não foram encontrados trabalhos que abordassem aspectos semelhantes aos do presente estudo.

Além disso, o filtro de ano de publicação no *Google Scholar* foi ajustado para o período de 2018 a 2023, com o intuito de assegurar que os resultados obtidos fossem baseados em pesquisas recentes e refletissem os avanços mais atuais na área. Dessa forma, garantiu-se que a revisão de literatura contemplasse os desenvolvimentos mais relevantes dos últimos cinco anos. Com essas buscas foram encontrados 96 artigos, dos quais apenas 4 possuem aspectos relacionados diretamente ao trabalho apresentado nesta monografia. Para a seleção desses trabalhos, foram considerados três fatores principais:

i) A integração de sistemas ou criação de sistemas utilizando API.
ii) A aplicação desses sistemas na área da saúde.
iii) Sistemas da saúde colaborando com outros sistemas.

Os demais 92 artigos foram descartados por não estarem em conformidade com os três fatores citados acima.

3.2 Sistema Integrado de Socorro para Idosos com Detecção de Quedas e Roteamento por BIM

Este trabalho apresenta uma estrutura integrada de um sistema de primeiros socorros para idosos para detectar automaticamente atividades de queda de idosos em um ambiente interno, monitorar o estado em tempo real dos idosos e formular a rota de resgate com o apoio do ambiente *Building Information Modeling* (BIM).

Para a detecção de quedas, o método consiste com a identificação de objetos pessoais em cada quadro de vídeo, seguida pelo rastreamento desses objetos ao longo do tempo. Depois, é realizada a detecção do esqueleto para reconhecer a estrutura corporal de cada pessoa. Por fim, utiliza-se uma *Convolutional Neural Network* (CNN) para determinar se o idoso está caindo ou não.

Quando ocorre a detecção de uma queda de um idoso, a expectativa é de que seja gerada a rota de resgate interna correspondente, levando em consideração o princípio do caminho mais curto e os atributos do edifício. A tomada de decisão para o roteamento de resgate interno é realizada dentro do ambiente BIM, utilizando API para facilitar o acesso às informações e a realização de inferências. A API integra o sistema de primeiros socorros para idosos em ambientes internos, utilizando técnicas de visão computacional.

Os resultados do experimento revelaram uma alta precisão na detecção das atividades de queda, atingindo uma precisão de 94,1%. Além disso, o método proposto permite a avaliação dos requisitos de acessibilidade de diferentes pontos de saída, considerando a seleção de vários equipamentos de transporte de primeiros socorros. Ao propor uma estrutura para o sistema de primeiros socorros destinado aos idosos, este artigo também demonstrou como a implementação de visão computacional e técnicas BIM pode ser aplicada para abordar o problema das quedas entre os idosos, particularmente em um ambiente de lar de idosos (CHEN *et al.*, 2022).

3.3 Monitoramento Médico Inteligente por IoT em Cidades Inteligentes

Este trabalho propõe criar um sistema inteligente para ambulâncias usando tecnologias novas, como 5G e Internet das Coisas. O objetivo é auxiliar pacientes que precisam de ajuda durante o transporte de ambulância. O sistema integra sensores de monitoramento de saúde, permitindo a transmissão de dados vitais para o hospital, agilizando o diagnóstico e tratamento. O paciente pode solicitar uma ambulância e enviar detalhes de emergência para os contatos. Essa abordagem salva vidas ao otimizar o tempo e fornece assistência médica mais rápida.

A pesquisa mostrou a eficácia do monitoramento remoto dos parâmetros de saúde do paciente por dispositivos *Android* e IoT. A aplicação transmitiu informações importantes para o hospital, dados esses que são os dados vitais do paciente, tornando rápido o processo de atendimento médico. Aplicar o sistema proposto fez com que o tempo de resposta fosse menor do que o sistema tradicional que é constantemente efetuado pelas pessoas na forma mais comum sem uso dessas tecnologias, apenas com ligação por telefone. A capacidade do sistema de permanecer com o paciente em tempo real para monitoramento contínuo foi enfatizada. A instalação do sistema na ambulância tornou mais fácil enviar informações de emergência para o centro médico, permitindo que o paciente acompanhasse o caminho da ambulância. O sistema tem uma função importante: enviar mensagens de emergência para contatos próximos com informações médicas e localização. A possibilidade de expandir a aplicação com mecanismos de segurança para proteger dados de saúde foi sugerida. Esses resultados mostram o potencial do sistema para melhorar o atendimento médico e salvar vidas (POONGODI *et al.*, 2021).

3.4 COVID-SAFE: Sistema IoT para Monitoramento de Saúde Pós-Pandemia

Este estudo propõe uma solução baseada na Internet das Coisas para enfrentar os desafios de saúde pública durante pandemias, como a COVID-19. Ele apresenta uma estrutura composta por dispositivos IoT de baixo custo, um aplicativo móvel e algoritmos de aprendizado de máquina para análise de dados em tempo real. Essa solução permite o monitoramento remoto da saúde dos usuários, incluindo parâmetros como temperatura corporal e frequência respiratória, além de fornecer orientações sobre o distanciamento físico necessário para reduzir a propagação do vírus. Os experimentos realizados demonstram a eficácia do sistema na medição precisa da distância entre dispositivos e na previsão do risco de propagação da infecção. Além disso, a análise de energia e largura de banda revela diferentes demandas em cenários de transmissão de

dados, permitindo adaptação às necessidades específicas de diferentes ambientes. Comparado a outras soluções existentes, este sistema oferece uma abordagem abrangente e eficaz para o monitoramento contínuo da saúde e a mitigação do risco de exposição ao vírus (VEDAEI *et al.*, 2020).

3.5 Modelo Híbrido de CNN para Diagnóstico de Câncer de Pulmão com Tomografia e Dados IoT Vestíveis

Este trabalho visa melhorar o diagnóstico de câncer de pulmão utilizando uma abordagem híbrida que combina Redes Neurais Convolucionais e dados da Internet das Coisas Médicas. A CNN, chamada *LungNet*, processa imagens de tomografia computadorizada para classificar o estágio do câncer com alta precisão. Os dados dos dispositivos vestíveis, que monitoram vários sinais fisiológicos, são integrados ao processo de diagnóstico para fornecer uma visão mais abrangente da condição do paciente. Isso permite um diagnóstico mais confiável e conveniente, além de facilitar o acesso remoto ao serviço de diagnóstico. O artigo descreve a arquitetura híbrida, a combinação de dados *Internet of Medical Things* (IoMT) e recursos de imagem, e propõe um serviço de classificação de estágio de câncer de pulmão acessível de forma onipresente.

O *LungNet* é um sistema que combina CNN com dispositivos IoMT. Esses dispositivos formam um *Mobile Body Area Network* (MBAN), que fornece dados para auxiliar um classificador. Isso aumenta a confiabilidade e a precisão das previsões. Após uma classificação inicial, os nódulos detectados são segmentados e subclassificados com base em seu tamanho. O desempenho do *LungNet* é avaliado usando métricas como precisão, sensibilidade e especificidade. O sistema consiste em quatro etapas: aquisição de dados, processamento de dados, subclassificação do estágio do câncer com base no tamanho do nódulo e diagnóstico e tomada de decisão. Os dados são coletados por dispositivos IoMT e enviados a um servidor para treinar o *LungNet*, que é implementado e treinado em *MATLAB* e *TensorFlow*.

O sistema *LungNet* demonstrou resultados robustos na classificação de nódulos de câncer de pulmão. Superou outras redes, alcançando uma precisão geral de 96,81% e uma sensibilidade de 97,02%. Após a aplicação de critérios de eliminação, o número de falsos positivos foi reduzido para 3,35% em média. Uma característica distintiva do *LungNet* é sua capacidade de classificar os nódulos em 5 classes e 4 subclasses, destacando-se como uma abordagem única no campo. Apesar das limitações, como dependência de dados MBAN e

necessidade de imagens aumentadas, o *LungNet* apresenta promessa para o desenvolvimento de sistemas automáticos de diagnóstico de câncer de pulmão amplamente acessíveis e confiáveis (FARUQUI *et al.*, 2021).

3.6 Comparação entre os Trabalhos Relacionados

Conforme apresentado na Tabela 1, todos os trabalhos possuem a implementação de um sistema na área da saúde e fazem uso de APIs para a integração entre sistemas de saúde ou para a utilização de recursos necessários ao funcionamento da aplicação de saúde implementada no trabalho em questão. Diferente de todos os outros trabalhos, apenas o trabalho apresentado na subseção 4.1 (1a linha da Tabela) não faz uso de sistemas baseados em IoT.

O diferencial deste trabalho, apresentado nesta monografia, em relação aos demais, é a aplicabilidade da API, que integra um sistema de detecção de quedas a um sistema de gestão hospitalar, com o sistema de detecção de quedas baseada em IoT.

Tabela 1 – Tabela de comparação entre os trabalhos relacionados e esta monografia

Trabalho	**Contexto do trabalho**	**Uso de API**
Sistema integrado de socorro para idosos com detecção de quedas e roteamento por BIM	Proposta de uma estrutura integrada que combina detecção de quedas baseada em visão computacional e roteamento de resgate acessível com suporte do BIM para melhorar a eficiência dos primeiros socorros a idosos	A API integra um sistema de primeiros socorros para idosos em ambientes internos, utilizando técnicas de visão computacional e modelagem de informação de construção (BIM)
Monitoramento médico inteligente por IoT em cidades inteligentes	Sistema de monitoramento de pacientes e rastreamento de ambulâncias em tempo real, utilizando IoT e 5G para enviar dados vitais ao hospital antes da chegada da ambulância, otimizando o diagnóstico e permitindo solicitações de emergência via aplicativo com acompanhamento de localização	A API conecta dispositivos de monitoramento de saúde, aplicativos de usuários e centros médicos
COVID-SAFE: Sistema IoT para monitoramento de saúde pós-pandemia	Apresenta a plataforma COVID-SAFE, que usa IoT para monitorar a saúde e alertar sobre o distanciamento físico durante a pandemia de COVID-19, integrando um dispositivo wearable, um aplicativo e um servidor para controle da disseminação do vírus	A API conecta o aplicativo de smartphone ao servidor, facilitando a comunicação na plataforma
Modelo híbrido de CNN para diagnóstico de câncer de pulmão com tomografia e dados IoT vestíveis	Apresenta o LungNet, um modelo híbrido que combina CNN e dados de sensores de Internet das Coisas (MIoT) para diagnóstico de câncer de pulmão. O objetivo é aprimorar a precisão na classificação dos estágios do câncer, integrando dados de tomografia e fisiológicos, propondo um serviço de Classificação de Estágio de Câncer de Pulmão	A API normaliza dados de dispositivos vestíveis e os armazena em um banco de dados relacional, combinando essas informações com imagens de tomografia para aprimorar o diagnóstico de câncer de pulmão
Esta Monografia	Utiliza dados de um sistema de detecção de quedas baseado em IoT para realizar a integração com um sistema hospitalar	A API envia as informações do usuário e informações de queda para um sistema hospitalar

Fonte: Elaborado pelo autor.

4 DESENVOLVIMENTO DA API

Neste capítulo são apresentados os passos de desenvolvimento da *FallReportAPI* proposta e implementada neste trabalho. Na Seção 4.1 é detalhado o processo de elicitação de requisitos para o desenvolvimento da API; e na Seção 4.2 é descrito todo o processo da construção da API desde a comunicação com o *Health Risk* até a chegada dos dados de queda no sistema do *Smart Hospital*.

4.1 Elicitação de Requisitos

No início do desenvolvimento deste trabalho, foi realizado um estudo e entendimento do código fonte disponível do projeto *Health Risk*, pois este já estava totalmente finalizado, enquanto o projeto *Smart Hospital* ainda se encontrava em desenvolvimento. Neste primeiro momento, o foco foi direcionado para as estruturas referentes ao paciente e em como o *frontend* se comunicava com o *backend* e vice-versa.

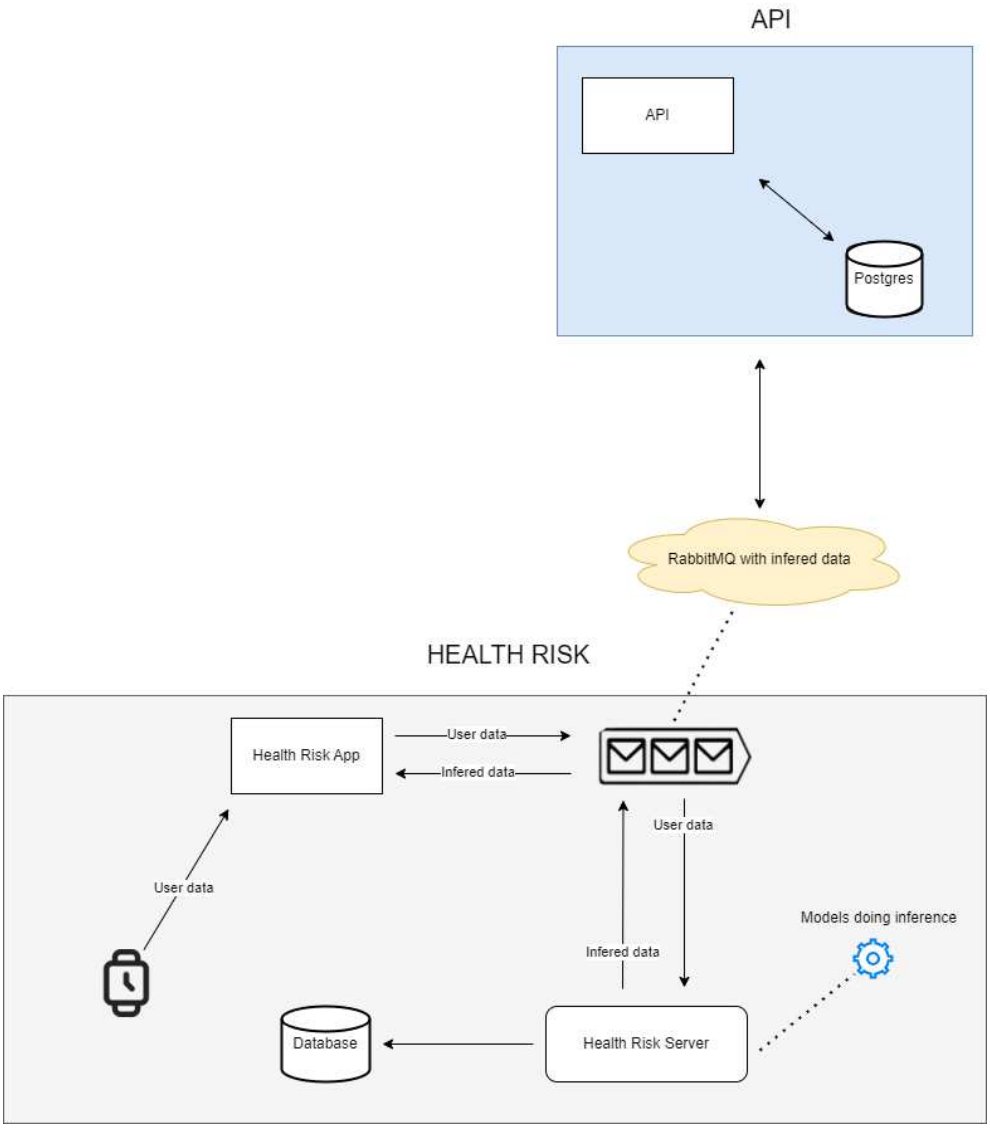
No contexto dos requisitos do projeto, foi realizado um estudo no código do sistema de detecção de quedas *Health Risk* e, ao analisar o código-fonte da aplicação, foram identificados os pontos de integração necessários para a construção da *FallReportAPI*. Além disso, ao examinar o fluxo de dados da aplicação, foi possível identificar os dados essenciais que podem ser integrados de forma consistente na API resultante. Focando nas informações pertinentes para a *FallReportAPI*, foi elaborado um documento de requisitos que especifica os dados a serem transmitidos para a API e consumidos pelo sistema do *Smart Hospital*. Este documento abrange exclusivamente os requisitos funcionais do sistema. Para o documento de requisitos foram considerados exclusivamente os requisitos funcionais da API. A consideração dos requisitos não funcionais poderá ser desenvolvida em projetos futuros.

Para maiores detalhes sobre o documento de requisitos elaborado para o desenvolvimento desta API, ele se encontra no Apêndice A desta monografia.

4.2 Projeto e Implementação da API

Como mencionado na Seção 4.1, os sistemas principais deste trabalho se encontravam em fases diferentes de desenvolvimento: uma aplicação totalmente finalizada, *Health Risk*, e outra aplicação que está ainda em sua fase de desenvolvimento, *Smart Hospital*. Para trabalhar com elas, foi preciso escolher a melhor abordagem para o início do desenvolvimento da *FallReportAPI*.

Figura 6 – Diagrama de integração



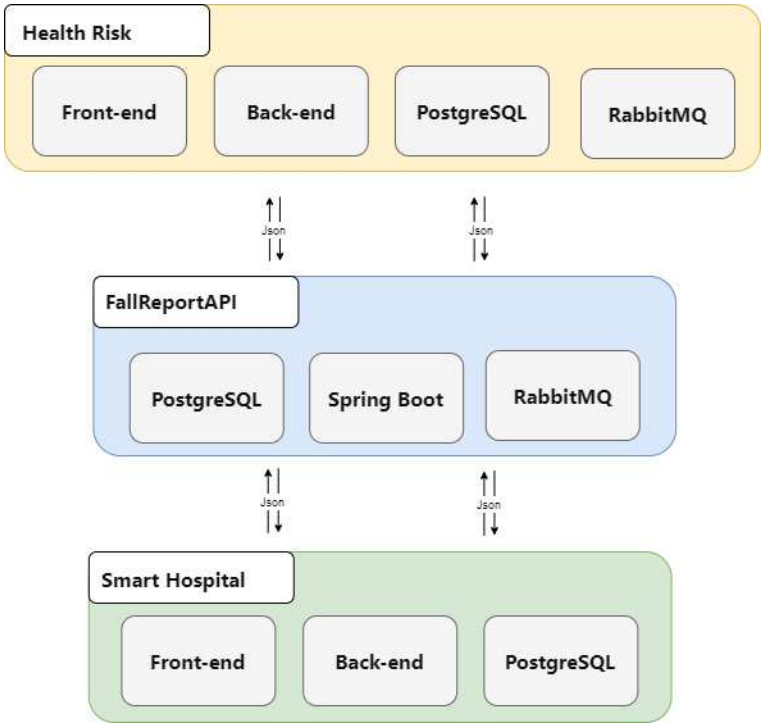
Fonte: Elaborado pelo autor.

Após a averiguação do código do *Health Risk* e tendo em vista que esta aplicação é a que enviará os dados principais que serão enviados para o *Smart Hospital*, é priorizado a integração, focando inicialmente na comunicação com a aplicação do *Health Risk* e, após finalizar essa etapa de comunicação, é feita as mudanças necessárias no projeto *Smart Hospital*, em sua forma mais madura, para receber os dados passados pela API.

Para ter uma base mais sólida do entendimento da comunicação da API com o *Health Risk* e um conhecimento mais aprofundado do código do *Health Risk*, foi feita uma figura que represente a comunicação do *Health Risk* com a API e uma figura que represente a arquitetura da integração dos sistemas *Health Risk* e *Smart Hospital*. A Figura 6 apresenta a integração entre a aplicação *Health Risk* e a API, enquanto a Figura 7 representa a arquitetura que será utilizada

como base para a implementação da API.

Figura 7 – Arquitetura da integração dos sistemas



Fonte: Elaborado pelo autor.

Após o entendimento de como o sistema *Health Risk* funciona, desde a estrutura do paciente, dos dados que são enviados pelo *Smartwatch* até o modelo de treinamento, é necessário partir para a construção da API propriamente dita. A linguagem de programação escolhida para o desenvolvimento foi *Java* utilizando o *framework Sprint Boot*. A escolha de trabalhar com a linguagem *Java* neste projeto se deu por ser uma linguagem de programação madura com vasta documentação para o desenvolvimento, além disso a linguagem garante uma estabilidade para projetos, auxiliando em um ciclo de vida longo. A combinação de *Java* e *Spring Boot* permite desenvolver APIs de forma rápida e eficiente, graças à sua configuração simplificada e às diversas ferramentas disponíveis. Aplicações construídas com *Spring Boot* podem ser facilmente escaladas para atender a demandas crescentes e o código organizado e modularizado, típico de aplicações *Spring Boot*, facilita a manutenção e o entendimento por outros desenvolvedores.

Com as Figuras 6 e 7 foi possível organizar a forma como a integração deveria acontecer por meio do sistema de mensageria que o *Health Risk* já estava utilizando, que é o *RabbitMQ*, neste caso utilizando o protocolo *Message Queuing Telemetry Transport* (MQTT) por meio do *RabbitMQ*. Como estratégia de desenvolvimento, optou-se por colocar a *FallReportAPI*

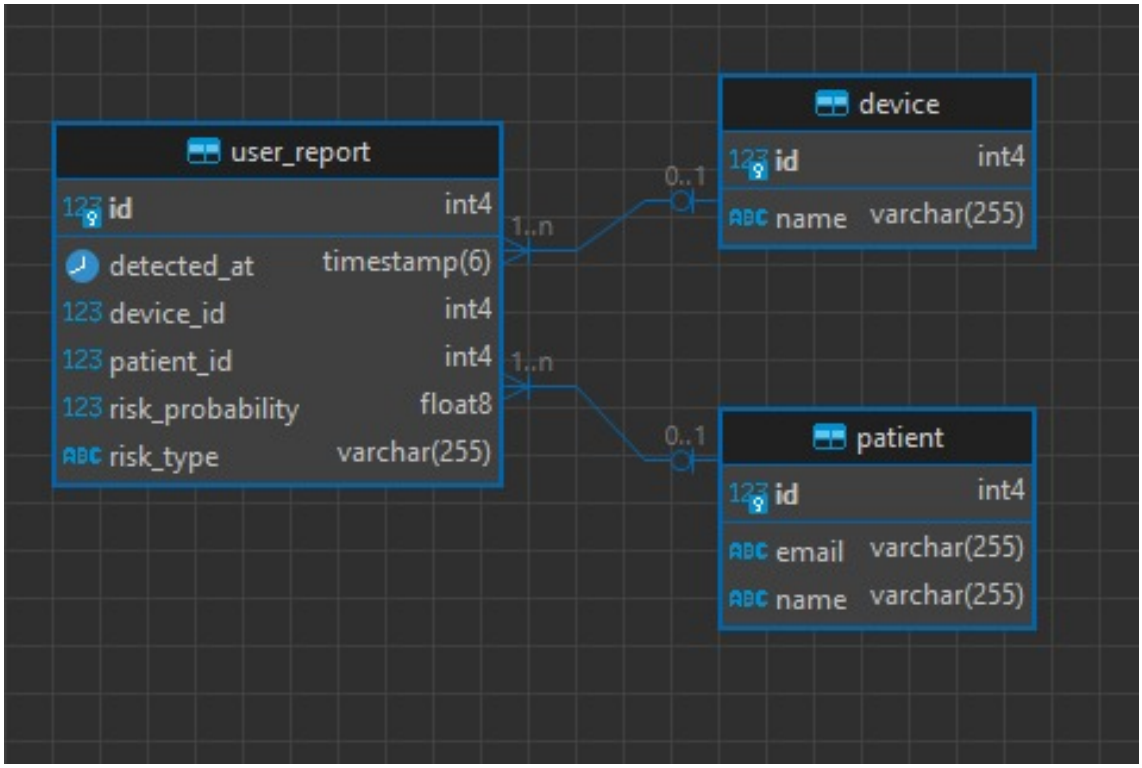
desenvolvida juntamente com um serviço consumidor via *RabbitMQ*; seria a melhor opção por encapsular a lógica de busca dos dados no banco de dados em um único serviço. Dessa forma não seria necessário criar as lógicas de busca no *backend* do *Smart Hospital*. A escolha de utilizar o *RabbitMQ* foi motivada pelo fato de que o sistema *Health Risk* já o adotava, permitindo aproveitar as filas existentes. A natureza assíncrona dos sistemas de mensageria também foi um fator determinante nessa decisão. Em relação à opção por uma API em vez de outras soluções, como microsserviços, inicialmente considerou-se a utilização do *backend* do *Smart Hospital* para realizar consultas no banco de dados, ao invés de criar uma nova API para esse propósito. Um microserviço seria responsável por persistir os dados obtidos via *RabbitMQ*. Contudo, a utilização do próprio *backend* para receber e encaminhar as notificações aos respectivos consumidores seria ineficiente, dada a quantidade potencial de notificações. Assim, a decisão de implementar uma API juntamente com o serviço consumidor visou promover a modularização, separando as rotas da API de integração das rotas do *backend* do *Smart Hospital*.

Para a construção da *FallReportAPI*, foi considerada a mensagem de relatório que indicava uma possível queda do paciente. Essa mensagem, que continha os dados necessários para o relatório, estava formatada em *JavaScript Object Notation* (JSON). Ela foi desmembrada em três modelos relacionados dos quais possuem essas informações: dados do paciente, dados do dispositivo, probabilidade de queda, *timestamp* referente ao momento da possível queda, e o tipo de risco.

A divisão dessa mensagem em três entidades que são *Device*, *Patient*, e *User Report* foi feita com a perspectiva de expansão do sistema. Dessa forma, cada uma dessas entidades poderá ser complementada com mais informações no futuro. Cada entidade possui uma camada de serviço dedicada. As entidades desenvolvidas e sua organização estão detalhadas na Figura 8. A *FallReportAPI* possui apenas três *endpoints*: um para buscar os relatórios de um paciente nas últimas 24 horas, com base no e-mail do paciente; outro para buscar todos os relatórios no banco de dados; e um terceiro para buscar todos os relatórios do banco de dados nas últimas 24 horas.

Concentrado-se na comunicação da *FallReportAPI* com o sistema do *Smart Hospital*, como a *FallReportAPI* foi desenvolvida separado do *backend* do *Smart Hospital*, não foi necessário modificar nada no *backend* do mesmo. As únicas modificações foram feitas diretamente no *frontend* do *Smart Hospital*. Essencialmente foi adicionado as rotas da API ao arquivo centralizado de rotas. Uma tela referente aos relatórios de queda foi criada no *frontend* do *Smart Hospital*. A ideia desse novo serviço no *frontend* é mostrar ao profissional da saúde

Figura 8 – Tabela do banco de dados da API



Fonte: Elaborado pelo autor.

informações de possíveis quedas de pacientes remotos, pacientes estes essencialmente usuários do *Health Risk*. É possível receber apenas informações de quedas das últimas 24 horas. Utilizar um intervalo de 24 horas para obter os dados dos usuários que sofreram uma possível queda é uma escolha sólida por várias razões. Primeiro, garante que os dados sejam recentes e relevantes, permitindo uma resposta rápida a eventos críticos, como quedas, o que é crucial para a segurança e o bem-estar dos usuários. Além disso, um intervalo de 24 horas é simples de implementar e monitorar, facilitando a análise dos dados. Também é possível buscar informações de queda de um paciente em específico com base no seu e-mail.

Vale ressaltar que neste trabalho é utilizado como dado único de busca a informação do e-mail do paciente, pois este é o único dado que dispõe no *Health Risk* como único por usuário e é um dado que também existe no banco de dados do *Smart Hospital* no cadastro do paciente. No entanto, é de conhecimento do planejamento do projeto para versões futuras que deverão ser acrescentados dados únicos mais comumente utilizados no ambiente hospitalar para identificação de pacientes, como o Cadastro de Pessoas Físicas (CPF) ou o Cartão do Sistema Único de Saúde (SUS), que são documentos diariamente utilizados nas unidades públicas de saúde do Brasil.

As Figuras 9 e 10 representam a tela criada no *Smart Hospital* para receber os

relatórios de quedas de usuários do *Health Risk*, sendo que a primeira tela apresenta todos os relatórios dentro do intervalo de 24 horas e a segunda tela apresenta os relatórios de queda de um determinado paciente identificado pelo e-mail. A Figura 11 representa a implementação da *FallReportAPI* se comunicando com *Health Risk* e repassando os dados para o *Smart Hospital*, ou seja, a API em sua forma completa.

Figura 9 – Tela do *Smart Hospital* que recebe relatório de queda

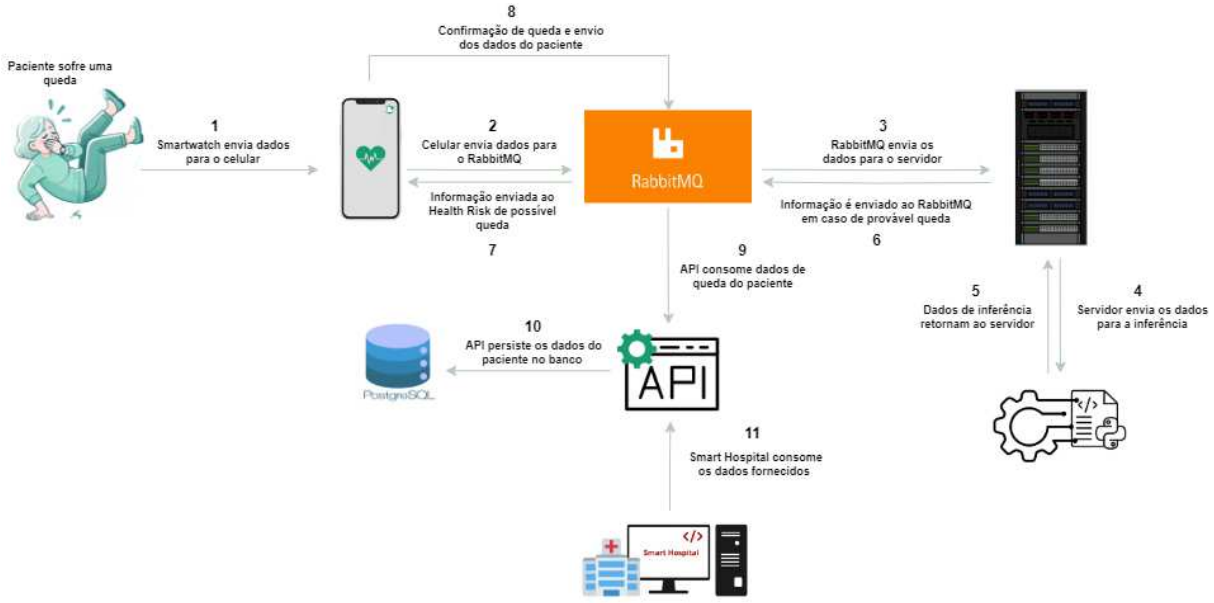
Fonte: Elaborado pelo autor.

Figura 10 – Tela do *Smart Hospital* que filtra o relatório por e-mail do paciente

Fonte: Elaborado pelo autor.

Importante ressaltar que para povoar dados no banco de dados do sistema do *Smart Hospital* foi criada uma conta na aplicação do *Health Risk* conforme representado pela Figura 3 e foram utilizados dados reais com o auxílio de um relógio *Smartwatch* da marca *Samsung*, o relógio em questão trata-se de um *Galaxy Watch4*. Além disso, para obter mais usuários de forma rápida, também foram cadastrados usuários via banco de dados pelo sistema *DBeaver*. O código de implementação desta API está disponível exclusivamente em um repositório privado do projeto devido à confidencialidade dos projetos envolvidos.

Figura 11 – Integração dos sistemas *Health Risk* e *Smart Hospital*



Fonte: Elaborado pelo autor.

Concentrando-se na Figura 11 que representa a implementação da *FallReportAPI* e a sua comunicação com os demais sistemas envolvidos, é possível observar os onze passos que representam essa comunicação entre os sistemas envolvidos e a API.

- Passo 1: O *Smartwatch* envia os dados coletados para o celular.
- Passo 2: O celular recebe os dados enviados pelo *Smartwatch* e os envia para o *RabbitMQ*.
- Passo 3: O *RabbitMQ* envia os dados recebidos para o servidor.
- Passo 4: O servidor envia os dados recebidos para os algoritmos de *Machine Learning* que farão as inferências desses dados.
- Passo 5: Os algoritmos de *Machine Learning* retornam os dados de inferência ao servidor.
- Passo 6: O servidor envia para o *RabbitMQ* as informações de uma provável queda.

- Passo 7: O *RabbitMQ* envia para o *Health Risk* as informações de possível queda para o celular.
- Passo 8: Se o usuário confirmar um caso de queda com um pedido de ajuda pela notificação do celular, os dados são enviados para o *RabbitMQ*.
- Passo 9: O serviço da API consome os dados de queda do paciente, pelo *RabbitMQ*.
- Passo 10: O serviço da API salva no banco de dados os dados do paciente.
- Passo 11: O *Smart Hospital* consome os dados do usuários através das rotas da API.

5 AVALIAÇÃO DA API

Neste capítulo são apresentados os testes realizados para validação e verificação da *FallReportAPI* bem como os resultados obtidos com os mesmos. Na Seção 5.1, é apresentado o conceito de teste de API que será realizada para a verificação da implementação feita neste trabalho. Na Seção 5.2 é apresentada a elaboração dos testes do código da *FallReportAPI* e seus resultados. Por fim, na Seção 5.3, é apresentada uma discussão dos resultados obtidos e um comparativo com o fluxo de atendimento de sistemas de saúde sem o uso da *FallReportAPI*.

5.1 Teste de API

Teste de API é a avaliação da qualidade e funcionalidade de interfaces de programação, verificando se elas operam conforme o esperado e apresentam os requisitos esperados. Essa avaliação envolve a simulação de diversas interações com a API e a comparação dos resultados obtidos com os esperados. O objetivo do teste de API é garantir que as interfaces de programação estejam entregando os resultados esperados com eficiência. Para isso, são realizadas diversas solicitações à API, simulando diferentes cenários de uso, e as respostas são analisadas em busca de erros ou comportamentos inesperados. O teste de API é uma etapa essencial no gerenciamento de API, sendo importante para assegurar a qualidade e a confiabilidade da API. Esse teste auxilia na identificação de *bugs* e inconsistências antes que a API seja disponibilizada aos usuários finais (Astera, 2024).

Neste trabalho, optou-se por realizar testes funcionais para garantir a obtenção dos resultados esperados. O foco foi a verificação dos *endpoints*, assegurando que as solicitações sejam processadas corretamente e que as respostas retornadas sejam adequadas ao esperado.

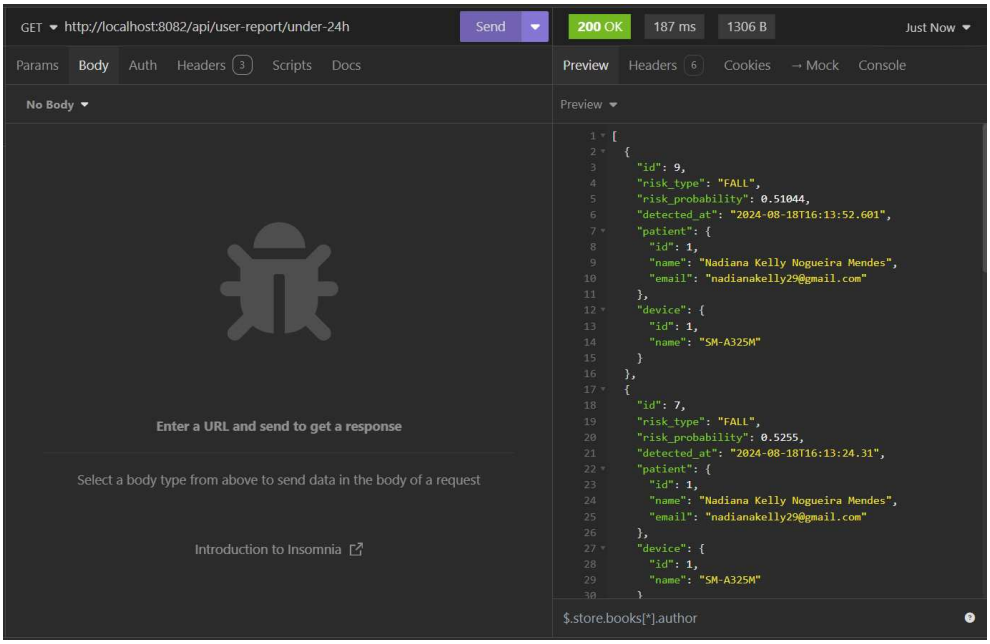
5.2 Testes Funcionais

Para a documentação do teste da API que integra os sistemas *Smart Hospital* e *Health Risk* foi elaborada uma documentação detalhada, cobrindo cada etapa do processo, incluindo a preparação, execução e resultados dos testes. A documentação completa do plano de testes da API proposta neste trabalho está detalhada no Apêndice B. Conforme apresentado na documentação de testes disponível no Apêndice B, é possível observar que a integração dos sistemas *Health Risk* e *Smart Hospital* foi realizada com sucesso. Não foram encontrados problemas na comunicação entre os sistemas por meio da *FallReportAPI*. Além disso, a *FallReportAPI*

executou a comunicação necessária entre os sistemas conforme esperado pela elicitação de requisitos.

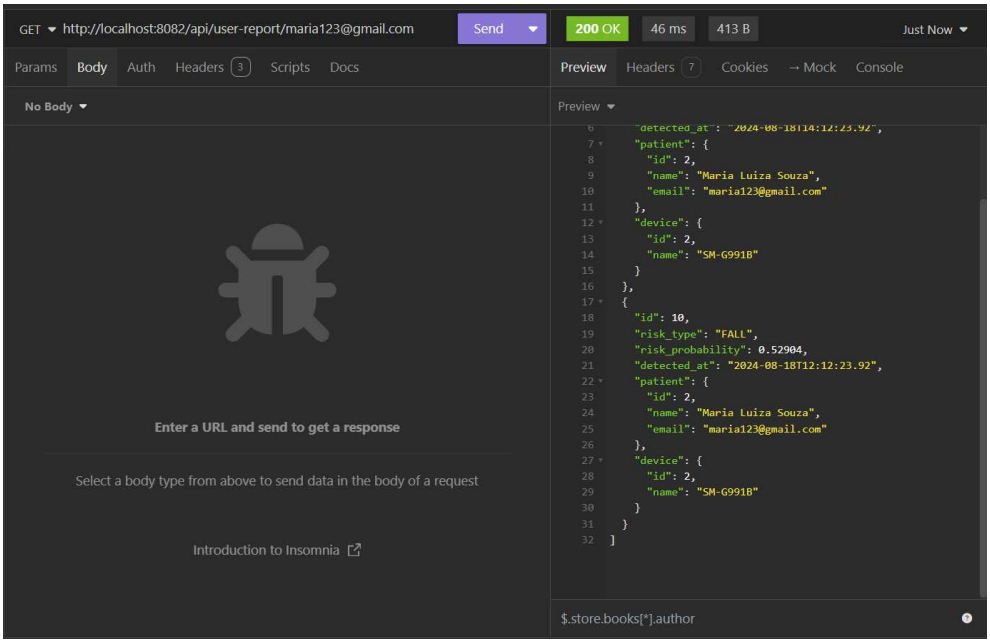
As Figuras 12 e 13 representam os testes executados para averiguar o comportamento da API de acordo com os requisitos exigidos.

Figura 12 – Teste de retorno de todos os usuários das últimas 24h



Fonte: Elaborado pelo autor.

Figura 13 – Teste de retorno de um usuário especificado por e-mail



Fonte: Elaborado pelo autor.

Para validar a integração entre o *Smart Hospital* e a *FallReportAPI*, foi necessário realizar testes funcionais tanto na API quanto na interface de usuário do sistema hospitalar. Inicialmente, os testes das rotas da API foram realizados utilizando o *Insomnia*, verificando se os dados de quedas detectadas eram enviados corretamente. Esses testes garantiram que a API estava funcionando conforme esperado, respondendo adequadamente às requisições e retornando os dados necessários. O ambiente de teste utilizado para avaliar a API consistiu em um dispositivo DELL i5 com o sistema operacional Windows 11 de 64 bits.

Após essa verificação, a segunda fase dos testes focou em garantir que os dados de quedas, transmitidos pela API, fossem corretamente recebidos, exibidos e manipulados pelo sistema hospitalar. Esses testes foram realizados diretamente pela interface, simulando a visualização dos relatórios de queda no sistema hospitalar, conforme os requisitos estabelecidos. A análise dos dados foi conduzida de acordo com os cenários especificados nos requisitos, e as telas criadas, como ilustrado nas Figuras 9 e 10, demonstraram que os dados foram corretamente processados e exibidos pelo *Smart Hospital*, obtendo um resultado satisfatório segundo os critérios de aceitação descritos no documento de requisitos, conforme apresentado no Apêndice A.

5.3 Discussão dos Resultados sem o Uso da API

A queda de idosos é considerada uma situação de urgência, especialmente se houver edema ou dor resultante, pois é necessário investigar a possibilidade de trauma (Secretaria da Saúde do Estado do Ceará, 2023). Um exemplo de que quedas também podem ser consideradas situações de emergência com risco à vida dos idosos é o fato de lesões causadas por quedas serem um dos atendimentos mais comuns em categorias de trauma. No município de Campina Grande, por exemplo, 3.271 pacientes com mais de 60 anos foram atendidos devido a quedas. Fraturas nos membros inferiores, causadas por quedas, apresentam um alto índice de mortalidade em pacientes idosos, uma vez que o paciente geralmente permanece acamado por longos períodos (Governo do Estado da Paraíba, 2023).

Utilizando essa análise do trabalho apresentado atrelada à informação de que queda de idosos é uma situação categorizada como urgência e emergência, faz-se necessário calcular a média de tempo esperado de um atendimento das unidades de saúde para esses casos para que possamos responder a questão de pesquisa **Q2**. Levando em consideração a definição da ISO/IEC 25010, que define a eficiência de desempenho como o desempenho em comparação com

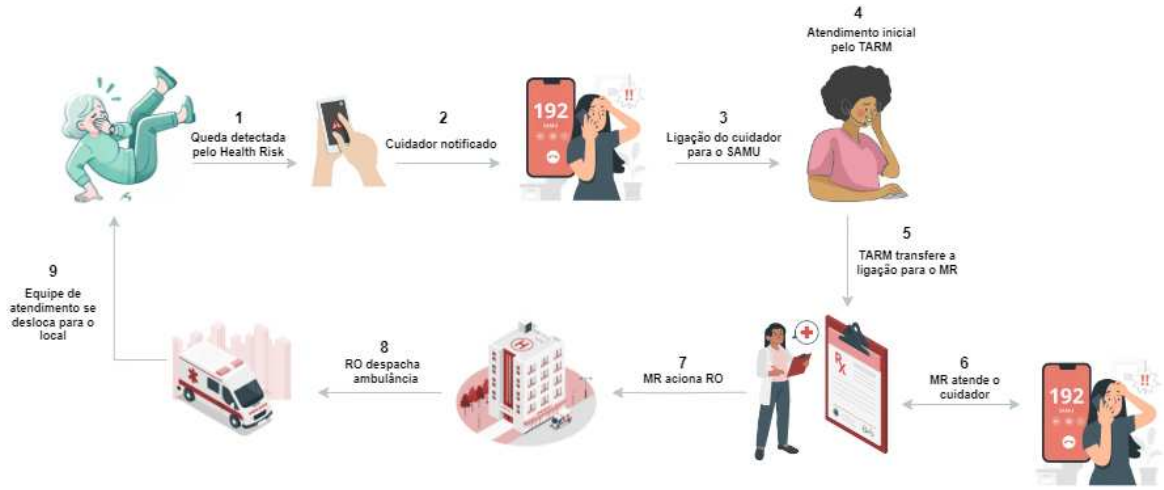
a quantidade de recursos empregados sujeito às condições determinadas (ISO, 2011), pode-se comparar a diferença de tempo de resposta entre o cenário de informação de queda passada ao hospital pela API com o tempo de resposta do cenário de informação de queda passada ao hospital por uma ligação de telefone.

Os serviços de ambulância são avaliados pelo tempo que decorre entre o recebimento de uma chamada de emergência e a chegada de um veículo ao local do paciente. Nesse sistema, todas as chamadas são classificadas em quatro categorias, que são elas: categoria de situação de risco de vida, categoria de situações de emergência, categoria de situações de urgência e categorias de situações menos urgentes. Concentrado-se nas categorias de emergência e urgência que são as categorias que podem englobar os cenários de quedas, principalmente com idosos, todos os fundos de ambulância devem responder a chamadas de categoria de emergência em 18 minutos em média, já para a categoria de urgência, os fundos de ambulância devem responder a 90% das chamadas em 120 minutos (Nuffield Trust, 2023). Dados recentes revelam que a média de espera do Serviço de Atendimento Móvel de Urgência (SAMU) para pacientes graves é muito maior do que a esperada. Segundo um levantamento recente de alguns estados do Brasil, o tempo médio de espera do atendimento do SAMU varia entre 15 minutos e 38 minutos (G1, 2019).

O tempo médio de atendimento do SAMU, de acordo com os dados mais recentes de 2022, varia em diferentes etapas do processo. Conforme representado na Figura 14, inicialmente, os Técnicos de Atendimento e Regulação Médica (TARM) realizam o atendimento das solicitações, com 59% dos casos sendo atendidos entre 1 e 2 minutos, 25% em menos de 1 minuto, 14% entre 2 e 3 minutos, e 2% levando mais de 3 minutos. Após o primeiro atendimento, a ligação é transferida para um Médico Regulador, que classifica a situação e decide o tipo de socorro. Este atendimento regulatório é feito em menos de 2 minutos em 46% dos casos, entre 2 e 4 minutos em 22% dos casos, e leva mais de 4 minutos em 32% das ocorrências. Após a regulação, o tempo para o despacho das viaturas pelo Regulador de Ocorrências é de menos de 1 minuto em 79% dos casos, entre 1 e 2 minutos em 9% dos casos, e mais de 2 minutos em 12% dos atendimentos. A equipe de atendimento começa a se deslocar para a ocorrência em menos de 2 minutos em 67% dos casos, entre 2 e 4 minutos em 16%, e mais de 4 minutos em 17%. O tempo de deslocamento da ambulância até o local da ocorrência é inferior a 5 minutos em 14% dos casos, entre 5 e 10 minutos em 29%, entre 10 e 20 minutos em 35%, e superior a 20 minutos em 22% das ocorrências. Considerando o tempo total desde o recebimento da chamada até a chegada da ambulância ao local, 11% dos atendimentos são feitos em menos de 5 minutos, 27%

em menos de 10 minutos, 36% entre 10 e 20 minutos, e 26% levam mais de 20 minutos (FILHO *et al.*, 2022).

Figura 14 – Fluxo de atendimento do SAMU



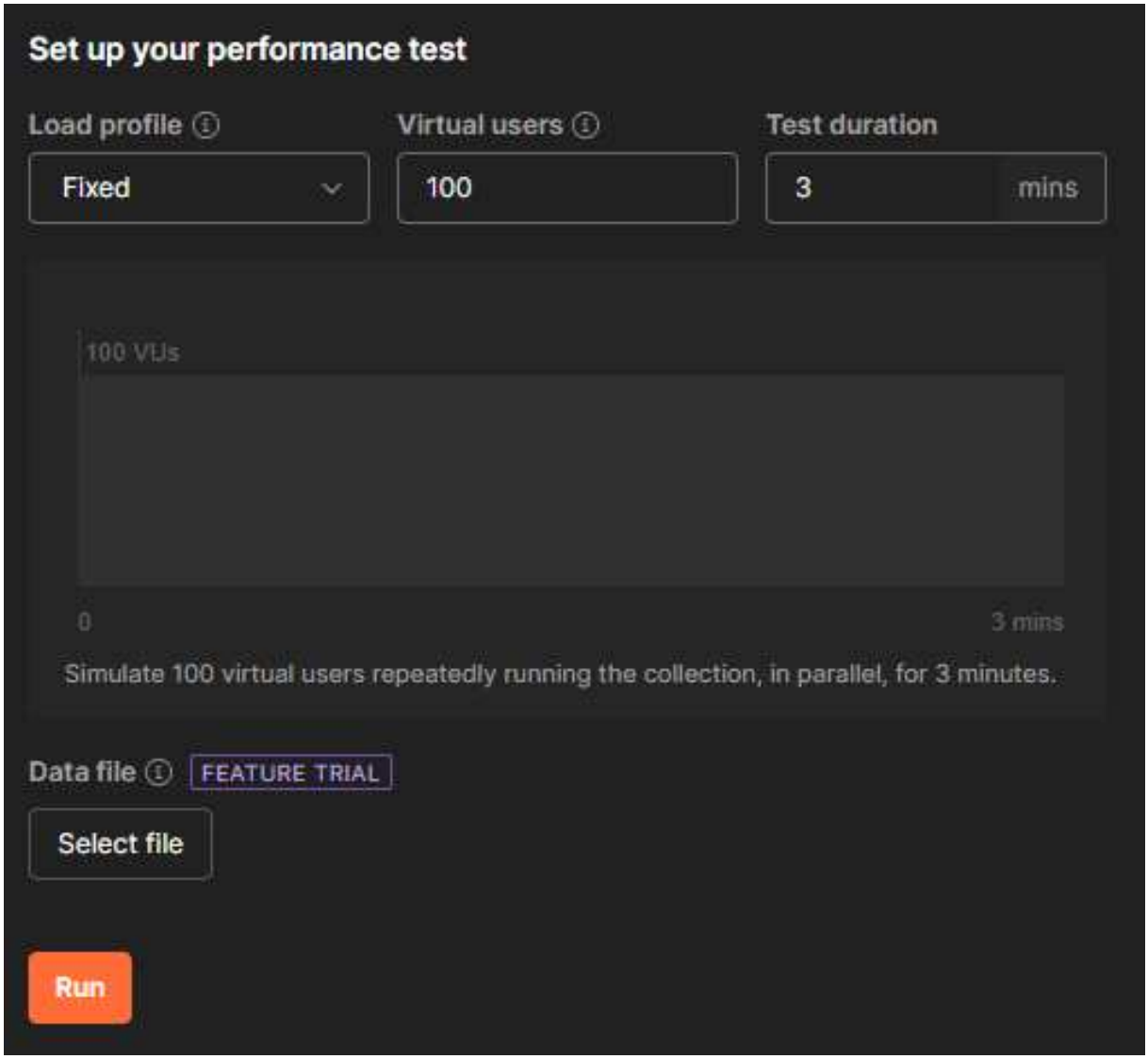
Fonte: Elaborado pelo autor.

É importante ressaltar a distinção entre o SAMU e outras ambulâncias disponíveis para atendimento, em síntese, o SAMU oferece atendimento pré-hospitalar com uma equipe médica especializada para lidar com casos de urgência, enquanto as ambulâncias são responsáveis pelo transporte de pacientes até hospitais ou unidades de saúde (SIQUEIRA, 2023). Outro fator importante é a diferença entre as situações classificadas como urgências e as que são consideradas emergências para a chamada de ambulâncias. As situações de emergência são aquelas que demandam avaliação rápida e podem necessitar de intervenção urgente no local ou transporte imediato. Já as situações de urgência referem-se a problemas que, embora não sejam imediatamente fatais, exigem tratamento para aliviar o sofrimento e podem necessitar de transporte ou avaliação clínica no local (Nuffield Trust, 2023).

Com o intuito de ter uma medida que comprove o bom desempenho da API em enviar as informações do usuário que sofreu uma queda, para o sistema desejado, neste caso o Smart Hospital, foi feito um teste de desempenho da API utilizando a ferramenta Postman, que possui a funcionalidade de teste de performance. O teste de performance feito na API analisou todas as três rotas da API. Conforme apresentado pela Figura 15, para obter o teste de performance da API foi configurado um teste com 100 usuários virtuais, com todos os usuários enviando requisições para a API durante o tempo de 3 minutos. A análise do desempenho da API está apresentada na Figura 16 mostra que a API tem um desempenho consistente, com tempos

de resposta médios variando entre 29 ms e 40 ms. O tempo de resposta médio geral é de 33,75 ms, calculado como a média dos tempos de resposta médios dos *endpoints* (40 ms, 34 ms, 32 ms e 29 ms). O tempo de resposta mínimo geral é de 7 ms, enquanto o tempo de resposta máximo geral é de 1.878 ms.

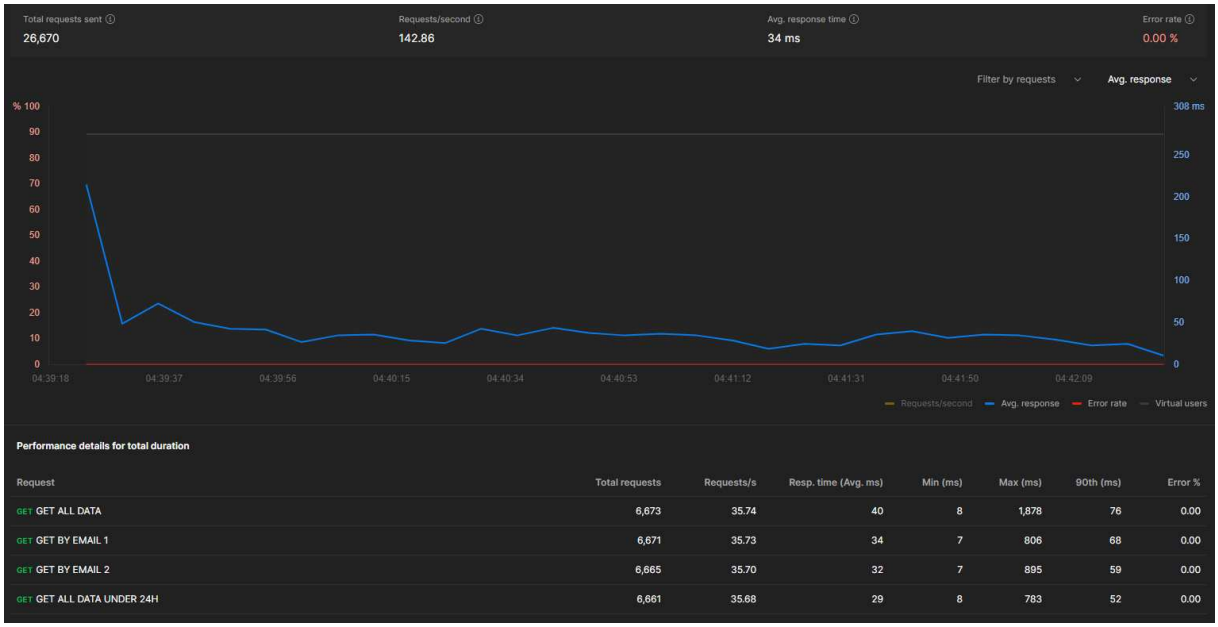
Figura 15 – Configuração do Postman para o teste de performance



Fonte: Elaborado pelo autor.

Os testes de desempenho da API foram realizados em um ambiente controlado. O ambiente de teste incluiu um dispositivo DELL i5 rodando Windows 11 de 64-bits, e a ferramenta utilizada para automatizar os testes foi o *Postman*. Esses testes visaram avaliar a capacidade da API de lidar com grandes volumes de dados e múltiplas requisições simultâneas. O foco principal foi em medir o tempo de resposta da API. Foram utilizados cenários que simulavam o uso da API, como o envio de múltiplas requisições de usuários caindo simultaneamente. Além

Figura 16 – Resultado do teste de performance com 100 usuários virtuais



Fonte: Elaborado pelo autor.

disso, foi testada a recuperação de dados de quedas em um intervalo de tempo mais extenso, para observar a capacidade de processamento da API.

Os resultados dos testes indicaram que a API manteve-se estável, com tempos de resposta dentro dos limites aceitáveis para a maioria dos cenários. Pequenos aumentos no tempo de resposta foram observados em casos de volumes muito elevados de requisições simultâneas, o que indicou a necessidade de melhorias na otimização para garantir uma performance mais uniforme. No entanto, não foram identificados problemas críticos que comprometessem o funcionamento da API em produção.

Concentrando-se no atendimento do TARM que coleta as mesmas informações que a API fornece, tendo o tempo registrado de 1 minuto e o maior tempo de 3 minutos por ligação telefônica, conforme mencionado anteriormente nesta seção, e que o atendimento a uma pessoa que sofreu uma queda pode ser classificado como uma situação de urgência ou emergência, dependendo do estado de saúde do paciente após o acidente, e sabendo que o SAMU é responsável pelo atendimento de urgência e emergência, espera-se que a comunicação da API dentro do fluxo de atendimento do SAMU, especificamente na fase de atendimento do TARM, possa melhorar a eficiência do processo de atendimento nas unidades de saúde que utilizem do Smart Hospital. Tendo em vista o desenvolvimento da *FallReportAPI* que integra o sistema do *Health Risk* ao sistema do *Smart Hospital* de tal forma que o hospital que utiliza o sistema do *Smart Hospital* recebe atualizações imediatas do estado do usuário do *Health Risk* em função

do seu monitoramento contínuo e com foco em situações de queda que necessitam de socorro, espera-se que os resultados obtidos com esta integração possibilite a comunicação mais rápida entre paciente e unidade de saúde.

Em resumo, a *FallReportAPI* atualiza a informação de queda de pacientes conectados ao *Health Risk* de forma ágil, enquanto com a abordagem de ligação telefônica, levando em consideração o impacto da API no tempo de atendimento do TARM com a ligação de um cuidador após notificação de uma queda da pessoa pela qual esse cuidador é responsável, é de no mínimo 1 minuto e no máximo 3 minutos. Então, considerando os resultados dos testes de desempenho apresentados acima, espera-se que, com o uso da *FallReportAPI*, esse atendimento possa ser reduzido consideravelmente, concentrando na resução no tempo de atendimento do TARM, pois com os dados fornecidos pela a API, a API gera um impacto de redução no tempo na etapa de atendimento do TARM. É notado que a eficiência de desempenho da API para agilidade de atendimento é superior ao atendimento por meios tradicionais, como é o caso de chamadas ao SAMU por telefone.

6 CONCLUSÃO E TRABALHOS FUTUROS

Este capítulo conclui esta monografia e está dividido da seguinte maneira: na Seção 6.1 são resumidos os principais resultados alcançados; na Seção 6.2 as hipóteses da pesquisa são revisitadas; na Seção 6.3 é apresentada as limitações deste trabalho; por fim, na Seção 6.4 são apresentadas as propostas de trabalhos futuros.

6.1 Resultados Alcançados

Este trabalho teve como objetivo desenvolver uma API denominada *FallReportAPI*, para a integração de dois sistemas de saúde: um sistema móvel de detecção de quedas baseado em Internet das Coisas e um sistema de gerenciamento hospitalar. Para o desenvolvimento da *FallReportAPI*, optou-se inicialmente por realizar uma revisão da literatura com foco em sistemas baseados em IoHT. Em seguida, foi conduzido um estudo detalhado das aplicações envolvidas na integração, especificamente os sistemas *Health Risk* e *Smart Hospital*. Esse estudo resultou na elaboração de uma documentação de requisitos, com ênfase nas informações que seriam transmitidas de um sistema para o outro, garantindo que pudessem colaborar entre si em situações de queda de pacientes.

A *FallReportAPI* foi desenvolvida utilizando Java como linguagem de programação, com o *framework Spring Boot* e *Hibernate* para a comunicação com o banco de dados SQL, e PostgreSQL para o armazenamento dos dados obtidos. A *FallReportAPI* foi submetida a testes de API e teste de desempenho, apresentou resultados satisfatórios, atendendo aos requisitos estabelecidos neste trabalho, com foco na integração eficiente e coerente entre os sistemas *Health Risk* e *Smart Hospital*.

6.2 Questões de Pesquisa

Para a questão de pesquisa **(Q1)**, com o desenvolvimento desta API, descrito no capítulo de desenvolvimento da *FallReportAPI*, é possível informar ao hospital as informações pertinentes sobre a queda de pacientes. O sistema *Smart Hospital* agora consegue receber, de forma ágil, dados sobre pacientes que sofreram quedas, permitindo um atendimento rápido aos usuários do *Health Risk* em unidades de saúde que utilizam o sistema *Smart Hospital*. Para a questão de pesquisa **(Q2)**, o impacto da integração desses dois sistemas de saúde é de natureza social, e espera-se que o uso da *FallReportAPI* traga benefícios à saúde ao possibilitar a redução

do tempo de atendimento. Isso ocorre porque a API envia os dados ao hospital, que, por sua vez, tem acesso às informações sobre a possibilidade e o risco de queda, auxiliando assim a equipe de atendimento. Além disso, espera-se que, com o uso da API, o fluxo do SAMU possa ser beneficiado com esses dados de queda e auxilie no fluxo de atendimento do TARM.

6.3 Limitações

A realização dos testes da API enfrentou algumas limitações significativas. Primeiro, não foi possível testar a API em cenários reais que envolvessem a sua integração completa com o sistema do Smart Hospital, uma vez que este ainda está em desenvolvimento no momento deste estudo. Além disso, a ausência de testes aplicados ao fluxo de atendimento do SAMU também representa uma limitação, pois não foi possível avaliar a performance da API em um ambiente de emergência real. Essas restrições limitam a abrangência dos resultados obtidos e sugerem a necessidade de estudos futuros para avaliar a eficácia da API em um ambiente real.

6.4 Trabalhos Futuros

Como trabalhos futuros, pode-se considerar os seguintes pontos:

- Implementação de Geolocalização: Integrar a coleta e envio de dados de localização para a API, permitindo que a localização exata do usuário que sofreu a queda seja enviada juntamente com as outras informações. Isso ajudará a direcionar serviços de emergência com maior precisão.
- Integração com o Fluxo de Atendimento Hospitalar: Estudar e propor maneiras de integrar o sistema de detecção de quedas ao fluxo de atendimento de urgência e emergência em hospitais, de modo que as equipes hospitalares estejam preparadas para encaminhar uma equipe de atendimento ao paciente, utilizando os dados fornecidos pela API.
- Monitoramento de Diversas Emergências de Saúde: Expandir a API para além da detecção de quedas, adicionando a capacidade de monitorar e enviar informações de outras emergências de saúde, como ataques cardíacos, crises epilépticas ou elevações críticas de pressão arterial, utilizando dados de diversos dispositivos de saúde conectados.

REFERÊNCIAS

ALMEIDA, R. L. A.; PAIVA, J. d. O. V.; GOUVEIA, T. N.; BARROSO, H. L. C. T.; NETO, J. B. F.; SANTOS, I. d. S.; EVANGELISTA, A. L. d. P.; ANDRADE, L. O. M. d.; BARRETO, I. C. d. H. C.; ANDRADE, R. M. C. Fictitious personas for interdisciplinary team alignment in the requirements elicitation activities. In: ACM. **Proceedings of the XVIII Brazilian Symposium on Software Quality (SBQS '19)**. [*S. l.*], 2019. p. 276–285.

ALVES, R. C.; COLICHI, R. M. B.; LIMA, S. A. M. *et al.* Tecnologias de monitoramento para prevenção de acidentes por quedas de idoso em ambiente hospitalar. **Pensamiento y Acción Interdisciplinaria**, v. 8, n. 2, p. 73–92, 2022.

ANDRADE, A.; PEREIRA, M.; SILVEIRA, S.; LINHARES, F.; NETO, A.; ANDRADE, R.; ARAÚJO, I. Continuous integration for machine learning experiments reproducibility: a practical study. In: SBC. **Anais do I Workshop Brasileiro de Engenharia de Software Inteligente**. [*S. l.*], 2021. p. 25–28.

ARAÚJO, I.; PEREIRA, M. B.; SILVA, W.; LINHARES, I.; MARX, V.; ANDRADE, A. M.; ANDRADE, R. M.; CASTRO, M. F. de. Machine learning and cloud enabled fall detection system using data from wearable devices: Deployment and evaluation. In: SBC. **Anais do XXII Simpósio Brasileiro de Computação Aplicada à Saúde**. [*S. l.*], 2022. p. 413–424.

ARAÚJO, I. L. de; JUNIOR, E. C.; DUARTE, P. A. d. S.; SANTOS, I. de S.; OLIVEIRA, P. A. M. de; MENDES, C. M. O.; ANDRADE, R. M. de C. Towards a taxonomy for the development of older adults healthcare applications. In: **HICSS**. [*S. l.*: *s. n.*], 2020. p. 1–10.

Astera. **The Ultimate Guide to API Testing**. 2024. Acesso em: 07 ago. 2024. Disponível em: https://www.astera.com/pt/type/blog/the-ultimate-guide-to-api-testing/.

AZARM, M.; BACKMAN, C.; KUZIEMSKY, C.; PEYTON, L. Breaking the healthcare interoperability barrier by empowering and engaging actors in the healthcare system. **Procedia computer science**, Elsevier, v. 113, p. 326–333, 2017.

BARRETO, I.; ANDRADE, L.; EVANGELISTA, A.; SANTOS, I.; ANDRADE, R.; OLIVEIRA, A.; NETO, J. F.; FROTA, V. Relato de concepção de aplicação para gestão da saúde da pessoa idosa dependente. In: SBC. **Anais do XX Simpósio Brasileiro de Computação Aplicada à Saúde**. [*S. l.*], 2020. p. 422–427.

BHATT, C.; DEY, N.; ASHOUR, A. S. **Internet of Things and Big Data Technologies for Next Generation Healthcare**. [*S. l.*]: Springer, 2017. v. 23. (Studies in Big Data, v. 23). Includes case studies of smart healthcare applications, such as telemedicine and health management applications.

CHEN, Y.; BOUFERGUENE, A.; SHEN, Y.; AL-HUSSEIN, M. Difference analysis of regional population ageing from temporal and spatial perspectives: A case study in china. **Regional Studies**, Taylor & Francis, v. 53, n. 6, p. 849–860, 2019.

CHEN, Y.; ZHANG, Y.; XIAO, B.; LI, H. A framework for the elderly first aid system by integrating vision-based fall detection and bim-based indoor rescue routing. **Advanced Engineering Informatics**, Elsevier, v. 54, p. 101766, 2022.

COSTA, A. F. F.; OLIVEIRA, V. T.; DANTAS, V. L. L.; SANTOS, I. D. S.; ANDRADE, R. M. D. C.; GOMES, R. L. Impacts of using pdca in the requirements specification process. In: **Proceedings of the XXII Brazilian Symposium on Software Quality**. [*S. l.*: *s. n.*], 2023. p. 244–253.

COSTA, C. A. d.; PASLUOSTA, C. F.; ESKOFIER, B.; SILVA, D. B. d.; RIGHI, R. d. R. Internet of health things: Toward intelligent vital signs monitoring in hospital wards. **Artificial Intelligence in Medicine**, Elsevier, v. 89, p. 61–69, 2018.

FARUQUI, N.; YOUSUF, M. A.; WHAIDUZZAMAN, M.; AZAD, A.; BARROS, A.; MONI, M. A. Lungnet: A hybrid deep-cnn model for lung cancer diagnosis using ct and wearable sensor-based medical iot data. **Computers in Biology and Medicine**, Elsevier, v. 139, p. 104961, 2021.

FILHO, H. L. A. F.; ARAUJO, C. M. F. de; JUNIOR, A. d. S. M.; FORASTIERI, H. L. C. *et al.* Tempo resposta no samu–192 e suas implicações. **Cadernos UniFOA**, v. 17, n. 49, p. 173–183, 2022.

FILHO, M. F. Internet das coisas. **Unisul Virtual**, 2016.

G1. **Samu demora mais do que o tempo recomendado para atender casos graves em 5 capitais**. 2019. Acesso em: 9 ago. 2024. Disponível em: https://g1.globo.com/ciencia-e-saude/noticia/2019/12/26/samu-demora-mais-do-que-o-tempo-recomendado-para-atender-casos-graves-em-5-capitais.ghtml.

Governo do Estado da Paraíba. **Lesões causadas por quedas são atendimentos mais comuns no trauma**. 2023. Acesso em: 09 ago. 2024. Disponível em: https://paraiba.pb.gov.br/noticias/lesoes-causadas-por-quedas-sao-atendimentos-mais-comuns-no-trauma-cg.

ISO. **ISO/IEC 25010: Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models**. 2011. Acesso em: 9 ago. 2024.

ISO. **ISO/IEC 25000:2014: Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – Guide to SQuaRE**. [*S. l.*], 2014. Acesso em: 16 set. 2024. Disponível em: https://www.iso.org/standard/64764.html.

JUNIOR, E. A. G.; ROCHA, R. D.; MACIEL, R. de S. Desenvolvimento de api rest com spring boot. **RIOS - Revista Científica do Centro Universitário do Rio São Francisco**, v. 15, n. 29, p. 499, 2021.

JUNIOR, E. C. **MOTION: Processo de desenvolvimento de aplicações de Internet of Health Things autoadaptativas baseadas em padrões de movimento**. Tese (Doutorado) – Universidade Federal do Ceará, Fortaleza, CE, 2023.

LINHARES, I.; ANDRADE, R.; JUNIOR, E. C.; OLIVEIRA, P. A.; OLIVEIRA, B.; AGUILAR, P. Lessons learned from the development of mobile applications for fall detection. **GLOBAL HEALTH**, v. 2020, p. 18–25, 2020.

MARTINS, C. I.; SILVA, K. R. da; CAMARGOS, M. C. S. Fatores sociodemográficos e clínicos associados ao atendimento de idosos vítimas de quedas em um pronto-socorro. **Revista de**

Administração Hospitalar e Inovação em Saúde, Universidade Federal de Minas Gerais, 2022.

NASCIMENTO, M. de M.; SILVA, P. S. T.; JUCHEM, L. Tecnologias assistivas: Aplicações na prevenção de quedas de idosos. **Saúde e Desenvolvimento Humano**, v. 10, n. 1, 2022.

NATIONS, U. World population ageing. **Department of Economic and Social Affairs**, p. 1–164, 2015.

Nuffield Trust. **Ambulance response times**. 2023. Acesso em: 9 ago. 2024. Disponível em: https://www.nuffieldtrust.org.uk/resource/ambulance-response-times.

POONGODI, M.; SHARMA, A.; HAMDI, M.; MAODE, M.; CHILAMKURTI, N. Smart healthcare in smart cities: wireless patient monitoring system using iot. **The Journal of Supercomputing**, Springer, p. 1–26, 2021.

RODRIGUES, J. J.; SEGUNDO, D. B. D. R.; JUNQUEIRA, H. A.; SABINO, M. H.; PRINCE, R. M.; AL-MUHTADI, J.; ALBUQUERQUE, V. H. C. D. Enabling technologies for the internet of health things. **Ieee Access**, IEEE, v. 6, p. 13129–13141, 2018.

ROSA, C. M.; SOUZA, P. A. R. d.; SILVA, J. M. d. Inovação em saúde e internet das coisas (iot): Um panorama do desenvolvimento científico e tecnológico. **Perspectivas em Ciência da Informação**, SciELO Brasil, v. 25, p. 164–181, 2020.

SANTOS, B. P. *et al.* Internet das coisas: da teoria à prática. In: **Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos**. [*S. l.*: *s. n.*], 2016. v. 31, p. 16.

Secretaria da Saúde do Estado do Ceará. **Quedas em idosos: entenda riscos, prevenção e saiba quando levar a uma emergência**. 2023. Acesso em: 09 ago. 2024. Disponível em: https://www.saude.ce.gov.br/2023/08/29/quedas-em-idosos-entenda-riscos-prevencao-e-saiba-quando-levar-a-uma-emergencia/.

SILVA, R. O. da; OLIVEIRA, J. L. S. de. A internet das coisas (iot) com enfoque na saúde. **Tecnologias Em Projeção**, v. 8, n. 1, p. 77–85, 2017.

SIQUEIRA, G. R. de. **Diferença entre SAMU e ambulância: entenda as principais distinções**. 2023. Acesso em: 7 set. 2024. Disponível em: https://brasilemergenciasmedicas.com.br/qual-a-diferenca-entre-samu-e-ambulancia/.

VEDAEI, S. S.; FOTOVVAT, A.; MOHEBBIAN, M. R.; RAHMAN, G. M. E.; WAHID, K. A.; BABYN, P.; MARATEB, H. R.; MANSOURIAN, M.; SAMI, R. Covid-safe: An iot-based system for automated health monitoring and surveillance in post-pandemic life. **IEEE Access**, v. 8, p. 188538–188551, 2020.

WEISER, M. The computer for the 21st century. **ACM SIGMOBILE mobile computing and communications review**, ACM New York, NY, USA, v. 3, n. 3, p. 3–11, 1999.

YUEHONG, Y.; ZENG, Y.; CHEN, X.; FAN, Y. The internet of things in healthcare: An overview. **Journal of Industrial Information Integration**, Elsevier, v. 1, p. 3–13, 2016.

APÊNDICE A – DOCUMENTO DE REQUISITOS DA API

INTRODUÇÃO

Este documento de especificação de requisitos tem como objetivo detalhar as necessidades e expectativas para o desenvolvimento de uma API que integrará o sistema móvel de detecção de quedas, *Health Risk*, com o sistema de gerenciamento hospitalar, *Smart Hospital*. A integração visa proporcionar uma comunicação eficiente e em tempo real entre os dois sistemas, permitindo que o *Smart Hospital* receba dados críticos sobre incidentes de queda de pacientes.

DEFINIÇÃO DE REQUISITOS DO USUÁRIO

A API fornecerá um serviço essencial para a integração entre o sistema móvel de detecção de quedas, *Health Risk*, e o sistema de gerenciamento hospitalar, *Smart Hospital*, pois espera-se que com o uso desta API o serviço de atendimento em casos de quedas seja mais ágil. A API realizará as seguintes funções:

i. Recepção de dados: A API receberá os dados importantes relacionados a incidentes de queda, que podem incluir informações do paciente e informações sobre o incidente ocorrido, no caso focando em situações de queda.
ii. Armazenamento de dados: Os dados recebidos serão armazenados em um banco de dados seguro e acessível.
iii. Consultas ao banco de dados: O sistema *Smart Hospital* poderá consultar o banco de dados através da API para acessar informações detalhadas sobre quedas de pacientes e tomar ações apropriadas para o atendimentos de urgência e emergência.

ESPECIFICAÇÃO DE REQUISITOS DO SISTEMA

Requisitos funcionais são especificações detalhadas que descrevem as funcionalidades e comportamentos esperados de um sistema ou produto. Eles definem o que o sistema deve fazer para atender às necessidades dos usuários. Os requisitos serão classificados em torno dos conceitos de requisitos funcionais. Este documento possui os requisitos funcionais necessários para o desenvolvimento atual da API. O foco neste momento é garantir que todas as funcionalidades essenciais e comportamentos esperados da API sejam claramente definidos. Embora este documento apresente uma análise do desempenho da API com o objetivo de embasar

a argumentação do estudo, a inclusão de requisitos não funcionais será considerada em etapas futuras, conforme necessário, para atender a critérios adicionais de qualidade. As Tabelas 2, 3 e 4 descrevem os requisitos funcionais para o desenvolvimento da API.

Tabela 2 – Tabela do requisito funcional RF001

Identificador	RF001
Nome do Requisito	Retorno dos dados de todos os usuários e dados relacionados a possibilidade de queda.
Descrição	A API deve fornecer todos os dados do usuário, conforme a tabela da seção de especificação dos atributos.

Fonte: Elaborado pelo autor.

Tabela 3 – Tabela do requisito funcional RF002

Identificador	RF002
Nome do Requisito	Retorno das informações de um usuário, incluindo dados relacionados à possibilidade de queda desse usuário, sendo o usuário identificado pelo e-mail.
Descrição	A API deve fornecer os dados do usuário, conforme a tabela da seção de especificação dos atributos, de acordo com o e-mail do usuário.

Fonte: Elaborado pelo autor.

Tabela 4 – Tabela do requisito funcional RF003

Identificador	RF003
Nome do Requisito	Retorno de todos os dados dos usuários em até 24h.
Descrição	Os dados exibidos na tela do *Smart Hospital* vão abranger um período de até 24 horas. No entanto, todos os dados serão armazenados no banco de dados, independentemente do tempo decorrido desde a queda.

Fonte: Elaborado pelo autor.

HISTÓRIA DE USUÁRIO

Histórias de usuário são uma forma simplificada de descrever como um sistema deve interagir com os usuários para realizar tarefas específicas, detalhando os passos necessários para alcançar um objetivo, focando nas necessidades e desejos dos usuários do sistema. As Tabelas 5 e 6 apresentam as histórias de usuário especificadas para o desenvolvimento da API.

Tabela 5 – Tabela da história de usuário H1

H1	Obtenção dos dados de todos os usuários
Requisito(s)	RF001 e RF003
Prioridade	Alta
Para quem?	Smart Hospital
O que?	Obtenção de todos os dados dos usuários cadastrados no *Health Risk* em caso de queda, conforme descrito na tabela de atributos.
Por que?	Porque o *Smart Hospital* precisa receber as informações sobre quedas de um usuário em tempo real para que possa tomar ações rápidas garantindo um atendimento de emergência ágil.
Critérios de aceitação	Fornecer todas as informações de acordo com a tabela de atributos de todos os usuários que sofreram quedas.

Fonte: Elaborado pelo autor.

Tabela 6 – Tabela da história de usuário H2

H2	Obtenção dos dados de um usuário específico
Requisito(s)	RF002 e RF003
Prioridade	Alta
Para quem?	Smart Hospital
O que?	Obtenção de todos os dados conforme descritos na tabela de atributos, de um usuário especificado pelo e-mail.
Por que?	Porque o *Smart Hospital* precisa receber as informações sobre quedas de determinados usuários.
Critérios de aceitação	Fornecer todas as informações de acordo com a tabela de atributos, de um usuário especificado pelo e-mail.

Fonte: Elaborado pelo autor.

ESPECIFICAÇÃO DOS ATRIBUTOS

Os atributos listados desempenham um papel fundamental na definição de como os dados são manipulados, exibidos e validados dentro da API:

a) **Obr.** - Preenchimento obrigatório.

b) **Aut.** - Preenchimento automático pelo sistema.

c) **Edit.** - Valor do atributo pode ser editado.

d) **Únic.** - Não aceita dados repetidos no banco de dados.

e) **Tipo:** Alfanumérico, Numérico, Data, Imagem, Botão, Link, Seleção única, Seleção múltipla, Autocomplete, Documento ou Filtro.

A Tabela 7 apresenta a especificação dos atributos relevantes para a API, detalhando como cada campo deve ser tratado em termos de obrigatoriedade, preenchimento, editabilidade, unicidade e tipo de dado. Esta tabela é essencial para o desenvolvimento e validação dos requisitos funcionais, garantindo que os dados sejam manipulados de maneira consistente e segura.

Tabela 7 – Tabela de especificação dos atributos

Título	**Descrição**	**Obr.**	**Aut.**	**Edit.**	**Únic.**	**Tipo (tam)**
Nome	Nome do usuário	S	S	N	N	String
E-mail	E-mail do usuário	S	S	N	S	String
Dispositivo	Nome do dispositivo celular do usuário	S	S	N	N	String
Tipos de risco	Tipo de risco do usuário, no caso focaremos nos riscos de queda	S	S	N	N	String
Probabilidade do risco	Probabilidade do risco que o usuário sofreu, no caso focaremos nos riscos de probabilidade de queda	S	S	N	N	Double
Data e hora	Data e hora que o usuário sofreu uma possível queda	S	S	N	N	DateTime

Fonte: Elaborado pelo autor.

APÊNDICE B – RELATÓRIO DE TESTE DA API

INTRODUÇÃO

Os testes desempenham um papel crucial no ciclo de desenvolvimento, permitindo identificar e corrigir defeitos. Sendo assim, esta documentação tem como objetivo descrever os procedimentos e as ferramentas utilizadas na realização dos testes para garantir a qualidade e o correto funcionamento da API desenvolvida.

REQUISITOS DA APLICAÇÃO A SEREM TESTADOS

Nesta seção, são apresentados os aspectos essenciais dos requisitos funcionais que precisam ser validados. A Tabela 8 fornece uma visão detalhada sobre os requisitos funcionais a serem testados.

Tabela 8 – Requisitos funcionais a serem testados e execução dos testes

Requisitos funcionais a serem testados	**Execução dos Testes**
Retorno dos dados do usuário de acordo com o e-mail em casos de queda nas últimas 24h	Manual
Dados de todos os usuários que sofreram queda nas últimas 24h	Manual

Fonte: Elaborado pelo autor.

ESTRATÉGIA DE TESTE

A Tabela 9 apresenta a estratégia de teste adotada para a API. A tabela descreve o nível de teste, o tipo de teste realizado, a abordagem empregada e a técnica específica utilizada.

Tabela 9 – Estratégia de testes

Nível	**Tipo**	**Abordagem**	**Técnica**
Integração	Funcional	Manual	Caixa preta
Performance	Não Funcional	Automático	Caixa preta

Fonte: Elaborado pelo autor.

AMBIENTE DE TESTE

A Tabela 10 descreve o ambiente de teste utilizado para a execução dos testes da API. A tabela detalha os principais componentes do ambiente, incluindo o método de execução, a ferramenta empregada, o dispositivo utilizado e o sistema Operacional em que os testes foram realizados.

Tabela 10 – Ambiente de teste

Execução	**Ferramenta**	**Dispositivo**	**Sistema Operacional**
Manual	Insomnia	DELL i5	Windows 11 64-bit
Automatizado	Postman	DELL i5	Windows 11 64-bit

Fonte: Elaborado pelo autor.

RECURSOS HUMANOS

A Tabela 11 apresenta os detalhes sobre os recursos humanos envolvidos no plano de teste. A tabela especifica o papel do responsável pelos testes, o número de Pessoas designadas para a função, as horas trabalhadas e as responsabilidades atribuídas. Para os testes da API, uma testadora foi designada para realizar uma carga de trabalho de 4 horas, com a responsabilidade de identificar não conformidades entre os requisitos estabelecidos e a API desenvolvida.

Tabela 11 – Recursos humanos

Papel	**Número de Pessoas**	**Horas Trabalhadas**	**Responsabilidades**
Testadora	1	4h	Buscar não conformidades entre os requisitos estabelecidos e a API desenvolvida.

Fonte: Elaborado pelo autor.

REGISTROS DOS DEFEITOS

Após a análise da API implementada em relação aos requisitos exigidos, não foram identificadas divergências. Durante os testes, não foram encontrados *bugs* na API.

RISCOS E CONTINGÊNCIAS

A Tabela 12 apresenta a análise de risco associada ao desenvolvimento da API. A tabela identifica o risco de entrega com *bugs*, classificado como de alto impacto. A estratégia de contingência descrita para mitigar esse risco envolve a revisão do plano de testes em colaboração com os desenvolvedores, com o objetivo de reduzir a quantidade de *bugs* no produto final. Esta abordagem é crucial para garantir a qualidade e a funcionalidade da API desenvolvida.

Tabela 12 – Análise de risco

Risco	**Impacto**	**Estratégia de Contingência**
Risco de entrega do projeto com bugs	Alto	Revisão do plano de testes juntamente com os desenvolvedores visando minimizar a quantidade de bugs no produto final.

Fonte: Elaborado pelo autor.

CASOS DE TESTE

As próximas tabelas detalham os procedimentos dos testes realizados para validar a funcionalidade da API desenvolvida.

A Tabela 13 apresenta o Teste CT-001, que avalia a obtenção dos dados de todos os usuários do *Health Risk* que possivelmente sofreram uma queda nas últimas 24 horas. A Tabela 14 descreve o Teste CT-002, que verifica a obtenção dos dados de um usuário específico do *Health Risk*, identificado por e-mail. A Tabela 15 detalha o Teste CT-003, que testa a resposta da API quando é solicitada a obtenção de dados de um e-mail inexistente.

Esses testes são fundamentais para assegurar que a API está operando conforme o esperado e para identificar qualquer falha na gestão dos dados dos usuários.

TESTE DE DESEMPENHO

O teste de desempenho é um tipo de teste não funcional que avalia como uma aplicação se comporta sob diferentes condições de carga. Ele fornece várias métricas importantes para avaliar a performance de um sistema. Neste documento, o foco será no teste de desempenho para obter a métrica do tempo de resposta da API, que mede o tempo que a API leva para

responder a uma solicitação.

Com o teste de desempenho, foi possível obter os seguintes resultados:

- **Tempo de Resposta Médio Geral**: (40 + 34 + 32 + 29) / 4 = 33,75 ms
- **Tempo de Resposta Mínimo Geral**: 7 ms
- **Tempo de Resposta Máximo Geral**: 1.878 ms

Esses dados mostram que a API apresenta um desempenho estável, com tempos de resposta médios que variam de 29 ms a 40 ms.

Tabela 13 – Caso de teste CT-001

ID do Caso de Teste	CT-001
História de Usuário	H1 do Apêndice A
Descrição	Testar requisição GET para obter relatório de todos os usuários das últimas 24 horas
Entrada	Endpoint: GET /api/user-report/under-24h
Saídas Esperadas	• Status 200 OK • Corpo da resposta contendo os dados do relatório em formato JSON
Procedimentos de Teste	1. Realizar requisição GET no endpoint: /api/user-report/under-24h 2. Verificar se o status de resposta é 200 3. Validar que o corpo da resposta contém os dados do relatório no formato esperado: • Nome • Email • Dispositivo • Risco de queda • Data e hora
Pré-Condições	Nenhuma
Pós-Condições	• Status 200 OK • Corpo da resposta contendo os dados do relatório em formato JSON

Fonte: Elaborado pelo autor.

Tabela 14 – Caso de teste CT-002

ID do Caso de Teste	CT-002
História de Usuário	H2 do Apêndice A
Descrição	Testar requisição GET para obter os relatórios de um usuário específico das últimas 24 horas
Entrada	• Endpoint: GET /api/user-report/email • Parâmetro: email = maria123@gmail.com
Saídas Esperadas	• Status 200 OK • Corpo da resposta contendo os dados do relatório em formato JSON

Procedimentos de Teste	1. Realizar requisição GET no endpoint: /api/user-report/maria123@gmail.com
	2. Verificar se o status de resposta é 200
	3. Validar que o corpo da resposta contém os dados do relatório no formato esperado:
	  • Nome
	  • Email
	  • Dispositivo
	  • Risco de queda
	  • Data e hora

Pré-Condições	O e-mail maria123@gmail.com deve existir na base de dados.
Pós-Condições	• Status 200 OK • Corpo da resposta contendo os dados do relatório em formato JSON

Fonte: Elaborado pelo autor.

Tabela 15 – Caso de teste CT-003

ID do Caso de Teste	CT-003
História de Usuário	H2 do Apêndice A
Descrição	Testar requisição GET para um usuário que não existe no sistema
Entrada	• Endpoint: GET /api/user-report/email • Parâmetro: email = usuario123@gmail.com
Saídas Esperadas	• Status 404 Not Found • Corpo da resposta com uma mensagem de erro informando que o usuário não foi encontrado
Procedimentos de Teste	1. Realizar requisição GET no endpoint: /api/user-report/usuario123@gmail.com 2. Verificar se o status de resposta é 404 3. Validar a mensagem de erro no corpo da resposta
Pré-Condições	O usuário cujo o e-mail é usuario123@gmail.com não deve existir na base de dados
Pós-Condições	• Status 404 Not Found • Corpo da resposta com uma mensagem de erro informando que o usuário não foi encontrado

Fonte: Elaborado pelo autor.