UNIVERSIDADE FEDERAL DO PARANÁ

ANGELO GABRIEL LEITE TROMBIM
GUSTAVO BRITO KRUEGER MAIA
LARISSA VICTORIA DA SILVA NEVES
LETICIA PAUMER
PAULA HOINACKI SUCLA

SMARTCABINET: AUXÍLIO NA ADMINISTRAÇÃO DE MEDICAMENTOS

CURITIBA
2023

ANGELO GABRIEL LEITE TROMBIM
GUSTAVO BRITO KRUEGER MAIA
LARISSA VICTORIA DA SILVA NEVES
LETICIA PAUMER
PAULA HOINACKI SUCLA

SMARTCABINET: AUXÍLIO NA ADMINISTRAÇÃO DE MEDICAMENTOS

Trabalho de conclusão de curso apresentado como requisito parcial à obtenção do grau, no Curso de Tecnologia em Análise e Desenvolvimento de Sistemas, Setor de Educação Profissional e Tecnológica, da Universidade Federal do Paraná.

Orientador: Prof. Dr. Alessandro Brawerman

CURITIBA
2023

TERMO DE APROVAÇÃO

ANGELO GABRIEL LEITE TROMBIM
GUSTAVO BRITO KRUEGER MAIA
LARISSA VICTORIA DA SILVA NEVES
LETICIA PAUMER
PAULA HOINACKI SUCLA

SMARTCABINET: AUXÍLIO NA ADMINISTRAÇÃO DE MEDICAMENTOS

Monografia aprovada como requisito parcial à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas, do Setor de Educação Profissional e Tecnológica da Universidade Federal do Paraná.

Prof. Dr. Alessandro Brawerman
Orientador – SEPT/UFPR

Prof. Dr. Luiz Antonio Pereira Neves
SEPT/UFPR

Profa. Dra. Rafaela Mantovani Fontana
SEPT/UFPR

Curitiba, 03 de Julho de 2023.

Documento assinado eletronicamente por **ALESSANDRO BRAWERMAN**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 03/07/2023, às 22:28, conforme art. 1º, III, "b", da Lei 11.419/2006.

Documento assinado eletronicamente por **RAFAELA MANTOVANI FONTANA**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 03/07/2023, às 22:30, conforme art. 1º, III, "b", da Lei 11.419/2006.

Documento assinado eletronicamente por **LUIZ ANTONIO PEREIRA NEVES**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 03/07/2023, às 22:31, conforme art. 1º, III, "b", da Lei 11.419/2006.

A autenticidade do documento pode ser conferida aqui informando o código verificador **5726043** e o código CRC **B52C5BB6**.

AGRADECIMENTOS

A todos amigos, familiares e colegas que contribuíram com seu suporte e compreensão, não só durante o desenvolvimento desse projeto, mas em toda nossa trajetória acadêmica.

Um agradecimento especial a Paulo Roberto Sucla, que apesar de não poder estar presente nessa etapa de conclusão de curso, sempre acreditou e apoiou sua filha. Que esse momento sirva para honrá-lo por todo seu esforço em contribuir com a minha trajetória.

RESUMO

Anualmente no Brasil, observa-se um crescimento significativo no consumo de medicamentos por pessoas que necessitam de algum tipo de tratamento clínico. Simultaneamente a essa questão, a dificuldade de gerenciamento dos pacientes quanto ao uso de remédios diminui sua adesão ao tratamento, o que pode provocar sérios problemas, que vão desde agravamento da doença, intoxicações por ingestão inapropriada e até a morte prematura. Além dos problemas já citados, a falta de um bom gerenciamento pode elevar o custo do tratamento e provocar gastos desnecessários. Neste contexto, percebe-se a necessidade de uma nova maneira de lidar com questões relacionadas à gestão de medicamentos. Portanto, este trabalho propõe a criação do projeto Smartcabinet, que conta com um aplicativo *Mobile* e um armário inteligente, e tem como objetivo facilitar o controle da utilização de medicamentos pelo uso de lembretes audiovisuais que indicam o horário correto para ingerir a medicação e que podem ser gerenciados por um aplicativo de celular. Assim, é possível configurar via aplicativo os medicamentos utilizados e receber as informações necessárias quanto à gestão, progresso e erros, a partir da interatividade com o armário, que conta com sensores, alertas sonoros e visuais para que o usuário realize o tratamento de maneira adequada. Para o desenvolvimento deste projeto, foi aplicada a linguagem UML (Unified Modeling Language) no processo de modelagem, auxiliando na criação de diagramas. A programação do aplicativo *Mobile* foi desenvolvida a partir do *framework* React Native e da linguagem JavaScript. Na elaboração do armário inteligente, foi utilizada a plataforma de prototipagem eletrônica Arduino para a manipulação e programação dos componentes eletrônicos. O projeto também contou com as metodologias ágeis Scrum e Kanban para o gerenciamento e divisão de tarefas.

Palavras-chave: Medicamentos. Gerenciamento. Aplicativo. Arduíno. Armário. React Native.

ABSTRACT

Annually in Brazil, there is a significant growth in the consumption of medicines by people who need some type of clinical treatment. Simultaneously to this issue, the difficulty of managing patients regarding the use of medicines reduces their adherence to treatment, which can cause serious problems, ranging from worsening of the disease, intoxication due to inappropriate intake and even premature death. In addition to the problems already mentioned, the lack of good management can increase the cost of treatment and cause unnecessary expenses. In this context, the need for a new way of dealing with issues related to medication management is perceived. Therefore, this work proposes the creation of the Smartcabinet project, which has a mobile application and a smart cabinet, and aims to facilitate the control of medication use through the use of audiovisual reminders that indicate the correct time to take the medication and that can be managed by a mobile application. Thus, it is possible to configure the medications used via the application and receive the necessary information regarding management, progress and errors, based on the interactivity with the cabinet, which has sensors, sound and visual alerts so that the user performs the treatment properly. For the development of this project, the UML language (Unified Modeling Language) was applied in the modeling process, helping in the creation of diagrams. The Mobile application programming was developed using the React Native framework and the JavaScript language. In the elaboration of the intelligent cabinet, the Arduino electronic prototyping platform was used for the manipulation and programming of electronic components. The project also relied on Scrum and Kanban agile methodologies for managing and dividing tasks.

Key-words: Medicine. Management. Mobile. Application. Arduino. Cabinet. React Native.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
IDE	*Integrated Development Environment*
OMS	Organização Mundial da Saúde
RTC	*Real Time Clock*
UFPR	Universidade Federal do Paraná
UML	*Unified Modeling Language*
WBS	*Work Breakdown Structure*

SUMÁRIO

1 INTRODUÇÃO

O Brasil conta, segundo dados do Conselho Federal de Farmácia de 2021, com cerca de 349 indústrias de medicamentos espalhadas por todo seu território. É um dos dez países que mais consome e demanda medicamentos em todo o mundo (CONSELHO NACIONAL DE SAÚDE, 2005). Também, de acordo com o levantamento do AQVIA Institute, em 2015 o Brasil já ocupava 10ª posição mundial no ranking dos mercados farmacêuticos e conforme revela o relatório da AQVIA Institute, a projeção é de que o Brasil ocupe a 5ª posição do Ranking dos Mercados Farmacêuticos do mundo até o ano de 2025 (IQVIA INSTITUTE, 2021).

De acordo com a Associação da Indústria Farmacêutica de Pesquisa (Interfarma), a venda de remédios cresceu cerca de 42,6% entre os anos de 2012 e 2016, e em 2021 subiu 14,21% em relação ao ano de 2020 (SINDUSFARMA, 2022). Leonardo Régis, doutor em Toxicologia e professor do Departamento de Ciências Farmacêuticas da Universidade de São Paulo (USP), afirma que remédios têm sido tratados como bens de consumo e que seu uso inadequado ou indiscriminado pode causar diversos problemas (RÉGIS, 2016). Porém, a permanência de um tratamento em algumas situações ultrapassa o controle do paciente, como o não acompanhado pelos ACS (agentes comunitários de saúde), falta de acesso aos medicamentos e a elevada complexidade na manipulação das doses diárias (REMONDI, CABRERA, SOUZA, 2014).

A adesão pode ser compreendida como a utilização de pelo menos 80% dos tratamentos prescritos, observando horários, doses e tempo de tratamento. A falta ou baixa adesão ao esquema terapêutico prescrito ao paciente é considerada por alguns autores como um problema de saúde pública, e tem sido denominada de “epidemia invisível”, variando de 15 a 93% para portadores de doenças crônicas, com média estimada de 50%, dependendo do método empregado para a medida (HELENA, 2007; SOUZA; GARNELO, 2008; BLOCH; MELO; NOGUEIRA, 2008).

Com estes dados apresentados podemos notar que a gestão de medicamentos pode impactar na qualidade de vida dos brasileiros e que há necessidade de melhor administração no manuseio desses.

A fim de ajudar a resolver problemas relacionados à gestão de fármacos, a proposta deste trabalho é criar um armário inteligente capaz de armazenar medicamentos e notificar o usuário via aviso luminoso e sonoro dos horários de

ingestão. O projeto também contou com um aplicativo que permite o cadastro de remédios e de suas respectivas informações de uso, proporcionando maior segurança aos seus usuários.

1.1 OBJETIVOS

Nesta seção será apresentada a descrição do objetivo geral deste trabalho, assim como a listagem detalhada de seus os objetivos específicos.

1.1.1 Objetivo Geral

O objetivo deste trabalho é desenvolver uma solução capaz de auxiliar no gerenciamento de ingestão de medicamentos. O Smartcabinet trata-se de um armário inteligente conectado a um aplicativo móvel, que visa notificar seus usuários de suas rotinas de tratamento, reduzindo as adversidades geradas com essa má gestão.

1.1.2 Objetivos Específicos

Os objetivos específicos do trabalho são:

a) Realizar pesquisas a respeito da ingestão de medicamentos, sobretudo no Brasil;
b) Pesquisar soluções semelhantes ao tema proposto;
c) Analisar ferramentas e tecnologias para o desenvolvimento do projeto;
d) Construir exemplos com o Arduino para compreender melhor o funcionamento da tecnologia;
e) Prototipar telas do aplicativo;
f) Elaborar diagramas;
g) Desenvolver funcionalidade de cadastro de medicamento no aplicativo;
h) Desenvolver funcionalidade de cadastro de remédios nas gavetas;
i) Desenvolver funcionalidade de acesso ao histórico de ingestão de medicamentos do usuário;
j) Desenvolver Dashboard contendo informações sobre o tratamento do usuário;
k) Programar as funcionalidades de registro de abertura de gaveta e notificação sonora e visual nos horários agendados do armário;

l) Integrar o armário com o aplicativo;

1.2 ESTRUTURA DO DOCUMENTO

Este documento possui informações referentes ao desenvolvimento do projeto, dividido em capítulos. O segundo capítulo tem como objetivo apresentar a fundamentação teórica do sistema, a qual detalha de forma mais aprofundada os assuntos relevantes relacionados ao projeto e a parte técnica envolvendo construção do armário e desenvolvimento do aplicativo. Entre os assuntos tratados estão estatísticas relacionadas ao impacto na utilização de medicamentos de maneira incorreta. Já sobre assuntos técnicos são encontrados os materiais, ferramentas e tecnologias utilizadas. O terceiro capítulo descreve todos os processos e ferramentas utilizados para auxiliar no planejamento e na execução do projeto. O quarto capítulo apresenta o sistema desenvolvido. Por fim, o quinto capítulo traz as considerações finais deste trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Nas próximas seções, serão apresentadas as tecnologias necessárias para o desenvolvimento do sistema, entendimento dos componentes nele utilizados e as referências bibliográficas indispensáveis à pesquisa. Para uma melhor compreensão do trabalho e dos assuntos a ele associados, as seções foram divididas nas categorias tecnologia, aspectos eletrônicos e soluções similares.

2.1 ADMINISTRAÇÃO INCORRETA DE MEDICAMENTOS

Por volta de 1940, surgiram os fármacos, fundamentais para o tratamento de doenças infecciosas, mal-estares, e outras enfermidades. Apesar de todos os benefícios que o uso de medicamentos pode proporcionar, a má administração dos mesmos pode acarretar graves problemas à saúde, e até mesmo prejuízos financeiros para o estado e instituições (MELO; RIBEIRO; STORPIRTIS, 2006).

No Brasil, os medicamentos são os principais agentes responsáveis por intoxicações humanas. O último relatório do Sistema Nacional de Informações Tóxico-Farmacológicas (Sinitox) de 2017, indica que os fármacos são responsáveis por 27,11% das intoxicações humanas, e em alguns casos também por óbitos (TELES et al., 2013).

De acordo com Shalini S. Lynch, PharmD em Farmácia pela Universidade de Califórnia, a adesão corresponde ao grau que o paciente segue o esquema terapêutico prescrito. Entre as razões para a falta de adesão estão a dosagem frequente e esquemas terapêuticos complexos como o uso de múltiplos medicamentos que devem ser tomados várias vezes ao dia (LYNCH, 2022).

Dentre as causas mais comuns da má utilização de remédios podemos listar fatores associados aos pacientes, como personalidade, alfabetização e barreira linguísticas, além de complexidade do caso clínico, incluindo múltiplas condições de saúde, polifarmácia e medicamentos de alto risco. E também, fatores associados ao medicamento, como nomenclatura e embalagem (WORLD HEALTH ORGANIZATION, 2016).

Um estudo realizado com o objetivo de identificar problemas práticos que pessoas de idade avançada experimentam com o uso diário de seus medicamentos

indicou as seguintes dificuldades: abertura da embalagem, identificação do fármaco (texto da embalagem muito pequeno ou extenso), separação em unidades individuais (sachês, frascos, copos blister) e medição da quantidade correta (NOTENBOOM, 2014).

Em março de 2014, o Ministério da Saúde, a Fundação Oswaldo Cruz e a ANVISA publicaram o Documento de referência para o Programa Nacional de Segurança do Paciente, o qual menciona que, em média, 10% dos pacientes internados sofrem algum tipo de evento adverso e destes 50% são evitáveis. Esse estudo também foi conduzido em outros países, como Austrália, Inglaterra, Canadá, Nova Zelândia e Dinamarca (DE VRIES, 2008).

A falta de adesão ao tratamento pode acarretar diversos problemas como o agravamento da doença, diminuição da eficácia dos medicamentos, aumento do risco de reações adversas, mudanças desnecessárias no tratamento e aumento do número de exames, de prescrições e de internações hospitalares. Isso pode elevar os custos do tratamento, criar gastos desnecessários e onerar o sistema de saúde, bem como levar à incapacidade e à morte prematura do paciente (CONSELHO NACIONAL DE SAÚDE, 2005).

Outro ponto relacionado a má administração de remédios, são as consequências financeiras causadas ao governo e às instituições. É estimado um gasto de aproximadamente US$ 4.700 por evento adverso de medicamento evitável ou por volta de US$ 2,8 milhões, anualmente, em um hospital com 700 leitos. O custo anual de morbidade e mortalidade referente a erros na medicação, nos EUA, tem sido estimado em torno de US$ 76,6 bilhões (BERWICK & LEAPE, 1999; KOHN et al., 2001, ANDERSON, 2002).

Segundo uma análise feita pelo Ministério da Saúde de Brasília, grande parte das intervenções utilizadas atualmente para melhorar a adesão medicamentosa nos problemas crônicos de saúde são ineficientes e instáveis. Intervenções viáveis e a longo prazo se fazem necessárias, também é evidente a necessidade de intervenções centradas no paciente (MINISTÉRIO DA SAÚDE, 2016).

As intervenções do tipo sistema de lembrete de medicação têm grande potencial para resolver problemas relacionados ao esquecimento de utilização de medicamento. Essas intervenções podem ser aplicadas na forma de mensagem de texto, chamadas automatizadas e dispositivos como lembretes audiovisuais (TRAN,

2014). O desenvolvimento destas soluções oferece uma oportunidade para o uso de diferentes tecnologias.

2.2 ASPECTOS DE TECNOLOGIA

Nesta seção, são apresentadas as principais tecnologias que compõem o trabalho e seu respectivo funcionamento.

2.2.1 Frontend

O desenvolvimento Frontend é associado à interface de uma aplicação, onde quem estiver utilizando o sistema poderá interagir diretamente. É considerado importante por ser capaz de proporcionar uma boa experiência ao usuário, através de uma aparência agradável e segurança em sua usabilidade (TOTVS, 2021).

Em suma, para o desenvolvimento de uma interface gráfica, é necessário a combinação do HTML, CSS e Javascript, que compõem a tríplice de tecnologias básicas do Frontend. São elas:

- *Hypertext Markup Language: é* uma linguagem de marcação, cuja principal função é possibilitar a interpretação dos conteúdos inseridos pelos navegadores. Tem a capacidade de gerar apenas telas estáticas (KRIGER, 2020).
- *Cascading Style Sheets:* tem como atributo estilizar as páginas geradas em um HTML, criando visuais esteticamente mais agradáveis (KRIGER, 2020).
- Javascript: responsável por trazer dinamismo e interatividade ao site. É uma das linguagens de programação mais importantes no cenário de desenvolvimento, que inclusive possibilitou a criação de diversas bibliotecas e *frameworks* derivadas da mesma (KRIGER, 2020).

2.2.2 Backend

O desenvolvimento backend é todo procedimento realizado no lado servidor da aplicação, onde o usuário não tem acesso, garantindo assim segurança na persistência de dados e contra-ataques maliciosos. No desenvolvimento backend são realizados acessos ao banco de dados, implementações de API, validação de dados, definição de rotas e tratamento de requisições e implementação da lógica de negócio.

Para o desenvolvimento backend temos algumas linguagens e frameworks especializadas em atuar neste campo, são elas JAVA, PHP, Ruby on Rails, Node.js, C#, C++, Kotlin, Rust, Go, Delphi, entre outras (EWALLY, 2021).

2.2.3 *Stack* Javascript

A linguagem de programação Javascript é utilizada pela maioria dos sites atuais e navegadores modernos, através de interpretadores, com o objetivo de trazer, em um primeiro momento, funcionalidades dinâmicas a uma página Web (ANJOS, 2017). Dentre suas principais características de composição, destacam-se a de ser uma linguagem não tipada, leve, de alto nível, dinâmica e interpretada, conveniente para estilos de programação orientados a objetos e funcionais (FLANAGAN, 2011).

Com o crescimento do Javascript, surgiram bibliotecas e *frameworks* baseados nesta linguagem, sendo que algumas permitem o desenvolvimento backend da aplicação, como ocorre com o Node.js (KRIGER, 2020).

No desenvolvimento mobile, o React Native é um framework de código aberto, derivado do Javascript, originalmente desenvolvido pelo Facebook após o sucesso do React em desenvolvimento de plataformas Web. É utilizado para criação de interfaces de aplicativos nativos em Android e IOS. Nesse *framework*, são utilizados componentes nativos do sistema operacional (Android ou IOS) para o desenvolvimento das aplicações, ao contrário do React, que utiliza componentes Web (WU, 2018).

O ambiente de execução Javascript conhecido como Node.js permite a criação de aplicações do lado do servidor, independente da utilização de um navegador, para executar seu código. Isso é possível pois, apesar de ser uma linguagem interpretada, a *engine* conhecida como V8, criada pela empresa Google, compila todo o código Javascript para linguagem de máquina. Algumas das características importantes do Node.js que se destacam, é a interpretação e recepção de requisições em *single thread*, ou seja, a aplicação terá instância de um único processo para entrada e saída de informações, conhecidos como *event loops* (DÜÜNA, 2016).

Desenvolvido e mantido pela Microsoft, o Typescript está disponível no mercado desde 2012. Apesar de ser derivada do Javascript trouxe alguns recursos adicionais como a possibilidade da verificação ser feita em tempo de compilação. Essa

característica é conhecida como tipagem estática, que não é encontrada no Javascript. Além de agregar ao Javascript um sistema de módulos, classes e interfaces. Com a utilização do Typescript, é possível integrar qualquer módulo do projeto escrito na mesma linguagem. O código é transpilado em Javascript para ser executado no ambiente. O programa gerado será interpretado em JS em tempo de execução (LOPES, 2016).

A complexidade do Javascript, levou ao desenvolvimento de ferramentas que facilitam a criação de componentes no sistema. A linguagem traz a resolução das deficiências do Javascript, como a aplicação em larga escala (NANCE, 2014).

2.3 ASPECTOS DE ELETRÔNICA

Esta seção tem por objetivo apresentar os conceitos técnicos e teóricos a respeito das tecnologias e componentes utilizados na parte física do projeto.

2.3.1 Sistemas Embarcados

Um sistema é classificado como embarcado quando é dedicado a uma única tarefa e interage continuamente com o ambiente a sua volta por meio de sensores e atuadores (STUART, 2005). É baseado em microcontroladores ou microprocessadores no caso dos mais complexos, e chips para o processamento dedicado. Possui tamanho compacto, de baixo custo e *design* simplificado (BOSON TREINAMENTOS, 2015). São utilizados em projetos de sistemas eletrônicos, tendo como principal função a execução de uma aplicação, ou conjunto de aplicações, relacionadas como um único sistema, tendo embutido em si um programa que interage com o meio externo (EMBARCADOS, 2018).

Os sistemas embarcados podem ser definidos como dispositivos que funcionam como computadores, que contam com memória, processador, interface de entrada e saída, tendo como principal diferencial o desempenho de uma tarefa específica, como por exemplo, o micro-ondas (POZZEBOM, 2014).

2.3.2 Arduino

Arduino é uma plataforma para prototipagem eletrônica muito utilizada inicialmente no desenvolvimento de projetos. Foi criado pelos professores Massino

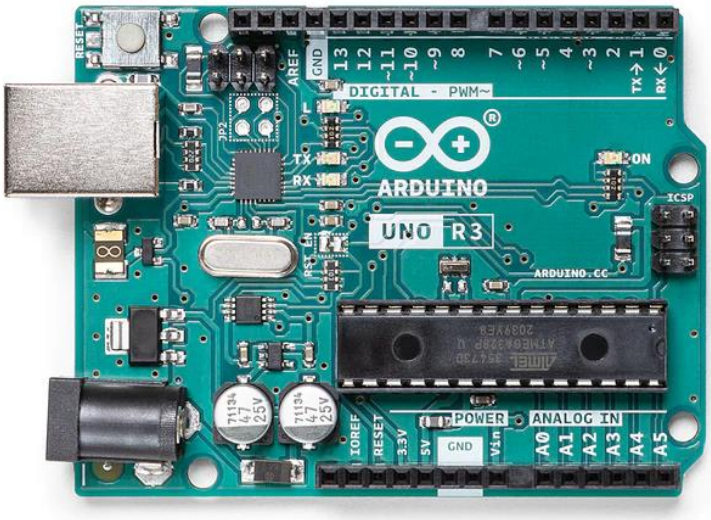
Banzi e David Cuartilles, em 2005, na Interaction Design Institute na cidade de Ivrea situado na Itália, com a missão de tornar o projeto de equipamentos eletrônicos viável para todos e mais simples do que outros sistemas disponíveis no mercado na época.

Trata-se de um dispositivo de sistema embarcado, por ter a capacidade de interagir com o ambiente através de hardware e software. É uma plataforma aberta (*open source*) de desenvolvimento constituída pela placa controladora (*hardware*) e ambiente de desenvolvimento (*software*). Permite que pessoas do mundo inteiro possam ter acesso a tecnologias avançadas que interagem com o mundo físico. Por ser flexível e de fácil acesso, tornou-se um dos embarcados mais utilizados tanto por estudantes para construção de projetos quanto por desenvolvedores profissionais (MCROBERTS, 2011).

O Arduino pode ser utilizado para desenvolver objetos interativos independentes, ou pode ser conectado a um computador, a uma rede, ou até mesmo à Internet para recuperar e enviar dados do Arduino e atuar sobre eles.

Na Figura 1, é possível observar a representação de um Arduino Uno R3.

FIGURA 1 - ARDUINO UNO



FONTE: ARDUINO.CC (2022).

Possui várias versões no mercado, sendo a versão mais recente o Arduino Uno (Figura 1), lançada em 2010. Se destaca por ser versátil e por isso, é atualmente o modelo mais usado e documentado. Esse modelo é uma placa baseada no microcontrolador ATmega328P. Esse componente apresenta tensão de operação de 5V, tensão de entrada recomendada de 7-12V, além de 14 pinos com entradas/saídas

digitais (GPIO, General Purpose Input/Output), dos quais 6 podem apresentar saídas com modulação por largura de pulso, e de 6 pinos analógicos (ARDUINO.CC, 2022).

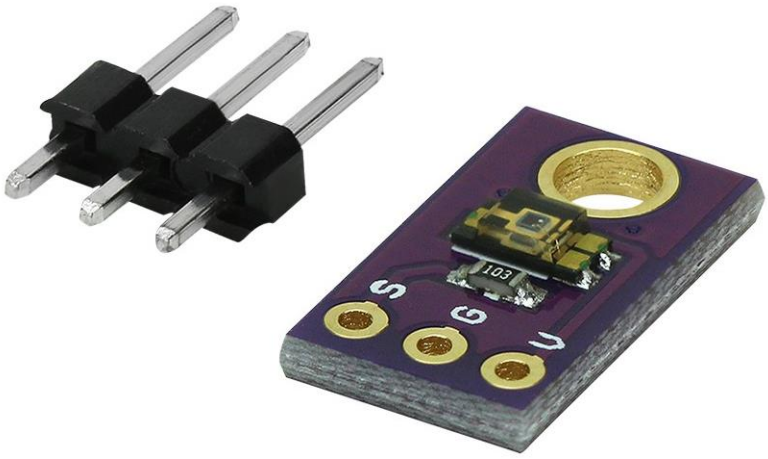
2.3.3 Sensores

Os dispositivos que, através do meio externo geram uma saída (normalmente uma tensão analógica) são conhecidos como sensores (PROTOTIPANDO, 2020). Já a associação de um módulo que transforma os estímulos a um sensor pode ser definida como transdutor (ELETROGATE, 2022).

As mudanças externas podem ser de qualquer tipo: de temperatura, gravidade, proximidade, luz, pressão, ou carga eletromagnética.

Na Figura 2, observa-se um exemplo de sensor utilizado em projetos no Arduino.

FIGURA 2 - SENSOR DE LUZ TEMT6000 USADO EM ARDUINO



FONTE: ELETRÔNICA FARIA.

2.3.4 Atuadores

Ao contrário dos sensores, atuadores convertem os sinais emitidos em quaisquer ações que o atuador exigir, ou seja, captam sinais elétricos e os modificam em algum tipo de saída física. Assim como os sensores, eles também são considerados transdutores. Exemplos comuns de atuadores são LEDs e Buzzers. O

primeiro converte energia e os estímulos elétricos em luz e o segundo em som (EMBARCADOS, 2019).

A Figura 3 demonstra um tipo de atuador conhecido como Buzzer, utilizado para produzir som.

FIGURA 3 - BUZZER

FONTE: GREEN INDUSTRYLINKS (2019).

2.3.5 Protoboard

*Protoboar*d é um componente eletrônico utilizado para simular placa mãe tendo *slots* de uma forma simples para plugar outros elementos contendo diversos furos para a montagens dos circuitos eletrônicos experimentais (LISBOA, 2015).

A principal vantagem de se utilizar uma *protoboard* é facilidade de acoplar os componentes sem a necessidade de solda.

Seu funcionamento consiste em linhas com pontos de alimentação com polos positivos e negativos, com ligações na horizontal. Já as linhas de implementação do circuito, ocorrem na vertical, ou seja, em colunas (LISBOA, 2015).

A Figura 4 ilustra uma protoboard com 4 conjuntos de filamentos para alimentação e 4 para a implementação do circuito.

FIGURA 4 - PROTOBOARD

FONTE: ELETROPEÇAS.

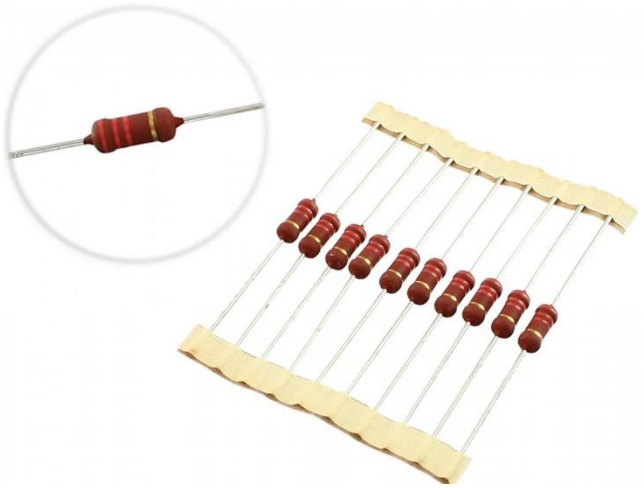
2.3.6 Resistores

Resistores são componentes eletrônicos utilizados para controlar a corrente elétrica e entregar apenas o necessário para não queimar as outras partes eletrônicas envolvidas. Ou seja, ele limita o fluxo de corrente elétrica dentro de um circuito (MUNDO DA ELÉTRICA, 2017).

Cada resistor possui um código de cores em seu corpo para o valor da resistência e sua tolerância. A primeira faixa indica o primeiro dígito do valor da resistência, a segunda listra o segundo dígito. A terceira é um fator multiplicador, e a quarta linha fornece o valor de tolerância do resistor (MUNDO DA ELÉTRICA, 2017).

Na Figura 5, é possível observar um resistor e as linhas em seu corpo indicando sua resistência e tolerância.

FIGURA 5 – RESISTOR



FONTE: USINAINFO.

2.4 SOLUÇÕES SIMILARES

Ao buscar soluções similares, foram encontrados projetos implementados em sistemas embarcados para controle de remédios. Estes projetos serão apresentados a seguir, com suas devidas utilidades e divergências referentes ao projeto.

2.4.1 MeDuino: SMART AUTOMATIC MEDICINE REMINDER

MeDuino é uma solução construída em Arduino com intuito de lembrar o usuário de sua a medicação diária. Seu objetivo é que através dos sons e luz emitidos pelo sistema, o usuário seja obrigado a chegar à caixa para acionar o botão que encerra os alertas gerados, lembrando-o de tomar o remédio (MINHAJ, 2018).

Trata-se de uma maleta simples de plástico, com Buzzer e LED acoplados, para emissão de alerta visual e sonoro. No entanto, qualquer configuração é feita diretamente no Arduino, ou seja, as notificações funcionam através de uma configuração de delay, sendo possível apenas alterar o espaço de horas entre as doses a serem tomadas (MINHAJ, 2018).

Por ser um projeto caseiro, a solução serve apenas para um nicho específico de usuários que têm acesso aos componentes necessários para o seu desenvolvimento, e que se limitam a tomar apenas um tipo de medicamento.

FIGURA 6 - SMART AUTOMATIC MEDICINE REMINDER

FONTE: ARDUINO.CC (2018).

2.4.2 LiveFine

LiveFine é uma solução que permite programar até nove alarmes por dia, tendo disponibilidade para carregar até 28 dias de medicação, todos controlados por seus botões centrais e, a cada horário programado, irá girar e disponibilizar o remédio do compartimento, emitindo aviso sonoro e visual, através de um display LCD (LIVEFINE, 2021).

Para evitar a ingestão desnecessária de medicamentos, o dispositivo conta com uma tampa bloqueável, que evita o acesso as doses pertencentes aos outros dias. Seu sistema consiste em liberar a divisão do dia atual nos horários programados. Apresenta como desvantagem, a possibilidade de que o usuário tenha acesso a todos os comprimidos do dia, e não só aos que ele precisa tomar em determinado momento (LIVEFINE, 2021).

Além disso, o sistema não possui controle via aplicativo, não disponibiliza informações sobre o progresso do tratamento do usuário, e sua disposição de compartimentos por dia pode ser um fator limitador na distribuição e controle dos medicamentos. E também, não possui conectividade com outros aparelhos para facilitar sua programação (LIVEFINE, 2021).

FIGURA 7 - DISPENSADOR DE MEDICAMENTOS AUTOMÁTICO LIVEFINE

FONTE: LIVEFINE (2021).

2.4.3 Caixa Time

Caixa Time é uma caixa automatizada com Arduino que permite o armazenamento de comprimidos em um horário pré-programado. Possui um alerta sonoro que apenas é silenciado quando a tampa é aberta (RODRIGUES, 2017).

A solução em questão trata-se de um sistema caseiro, onde o próprio usuário constrói a Caixa Time com o Arduino. Os itens necessários para a construção do dispositivo são: Uma caixa de primeiro-socorros, um Arduino Uno R3, um módulo RTC DS3231, um Buzzer, uma Chave fim de curso, uma fonte de 9v, um Resistor de 22 ohms e cabos tipo Jumper (RODRIGUES, 2017).

Essa solução não conta com programação via aplicativo e apenas disponibiliza cadastrar um intervalo de horário. Além de ser um sistema básico que não permite administrar múltiplos medicamentos.

2.4.4 Comparativo entre as soluções

A seguir, a Tabela 1 apresenta um comparativo entre os dispositivos encontrados que possuem uma proposta semelhante a proposta neste trabalho. Como é possível observar, todos os dispositivos possuem as funcionalidades essenciais

como agendamento e avisos, porém não apresentam diferenciais como conectividade com outros aparelhos, disponibilização de histórico e progresso do usuário. O SmartCabinet possuí todos os recursos descritos mostrando-se como a melhor opção em relação as opções similares encontradas.

TABELA 1 - COMPARATIVO DE DISPOSITIVOS

Recursos	MeDuino	Caixa Time	LiveFine	SmartCabinet
Agendamento	X	X	X	X
Aviso sonoro	X	X	X	X
Aviso visual	X		X	X
Múltiplos medicamentos			X	X
Conectividade com outro dispositivo				X
Histórico				X

FONTE: OS AUTORES (2023).

3 MATERIAIS E MÉTODOS

Ao decorrer do capítulo anterior foi apresentado a fundamentação teórica e técnica que embasam este projeto. A apresentação de estudos que demonstram o quão custoso é a má gestão de medicamentos, mostra a utilidade de se usar um administrador de remédios. O capítulo atual tem como objetivo descrever todos os processos e ferramentas que foram utilizadas para auxiliar no planejamento e execução do projeto para que os objetivos fossem atendidos.

3.1 MATERIAIS

Levando em consideração a disponibilidade de conteúdo das ferramentas e tecnologias, foram selecionados programas para o auxílio da organização do projeto, e para desenvolvimento da modelagem, *softwares* e *hardwares*.

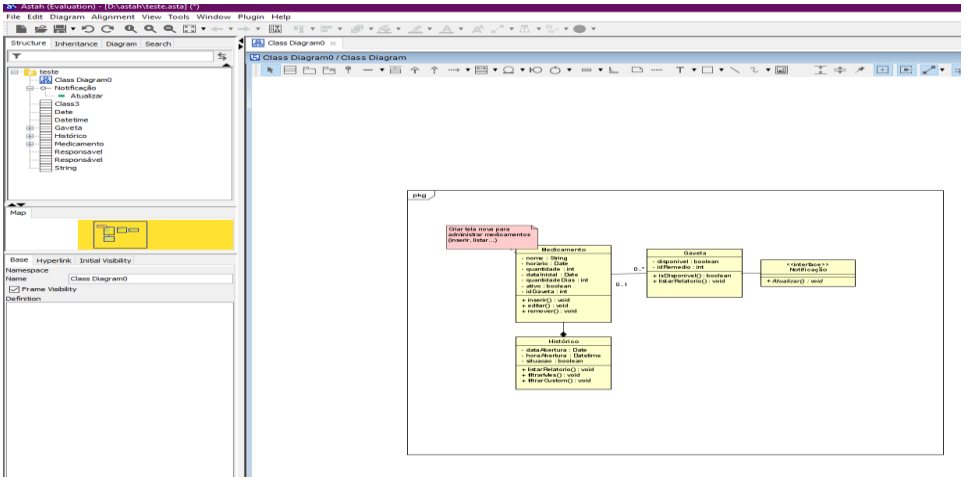
3.1.1 Astah

Uma documentação organizada é fundamental para o entendimento ideal do sistema. Ela traz segurança, menos dependência de profissionais e facilita questões relacionadas à comunicação, devendo ser parte da rotina da área de Tecnologia da Informação tanto para projetos de grande porte quanto projetos menores. A modelagem traz grande auxílio para definição de funcionalidades, componentização, fluxos e arquitetura de um sistema (ASTAH, 2022).

O software Astah Professional (ASTAH, 2022) foi utilizado pela equipe para elaboração de todos os diagramas UML, sendo escolhido por sua interface intuitiva e praticidade para gerar os devidos documentos.

A Figura 8 demonstra a plataforma Astah sendo utilizada para o desenvolvimento do diagrama de classes.

FIGURA 8 - FERRAMENTA ASTAH



FONTE: OS AUTORES (2023).

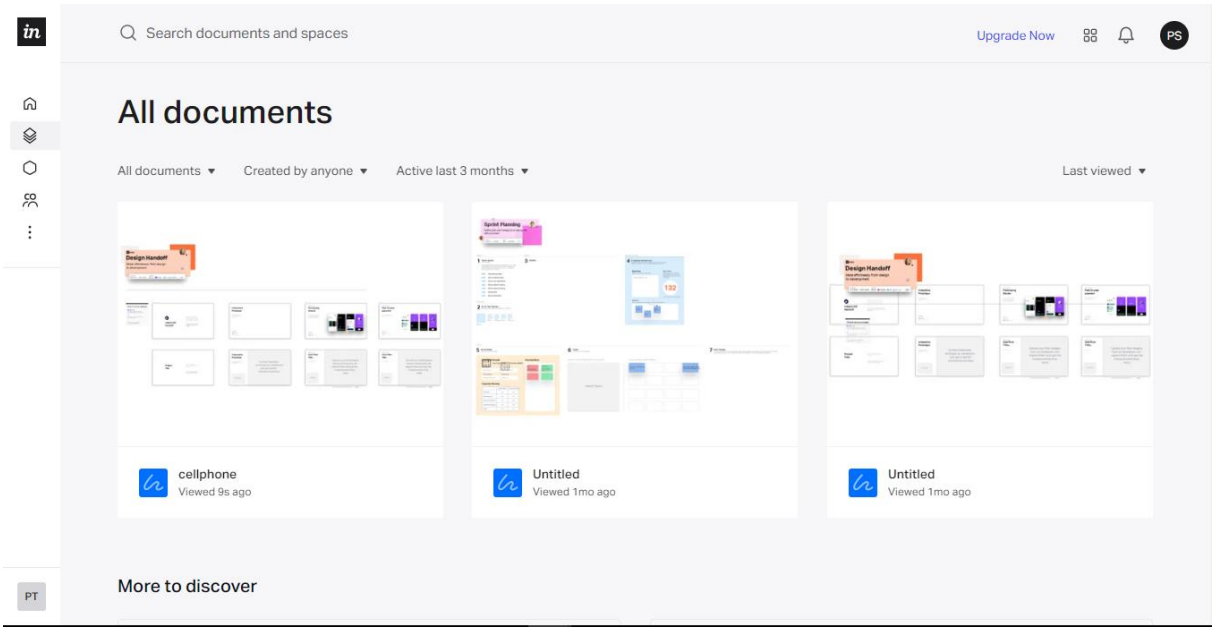
3.1.2 InVision

A prototipagem é essencial para a composição de um projeto pois permite descrever suas funcionalidades e comportamentos de forma que atendam os objetivos estabelecidos pelo escopo.

A equipe escolheu utilizar para a prototipação de telas o InVision, uma plataforma digital utilizada para a criação de protótipos, sendo popular por sua baixa curva de aprendizagem e possibilidade de apresentar dinamicamente o funcionamento de um projeto para o usuário.

Na Figura 9, pode-se observar a tela de projetos da ferramenta InVision.

FIGURA 9 - FERRAMENTA INVISION



FONTE: OS AUTORES (2023).

3.1.3 Trello

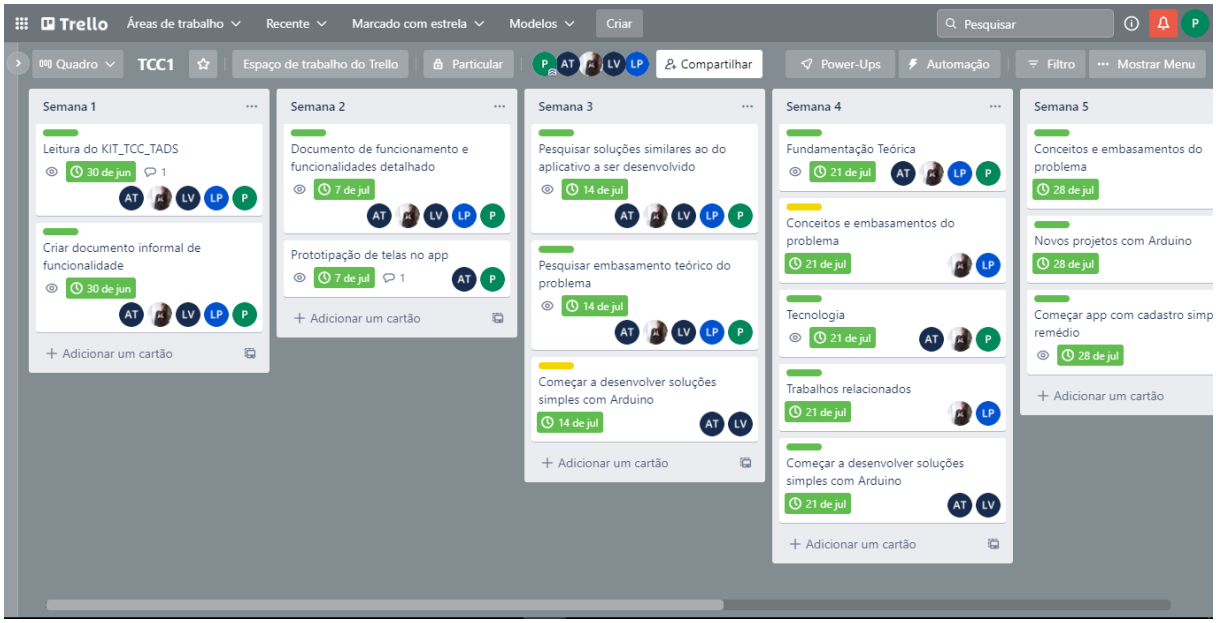
O Trello foi lançado inicialmente na TechCrunch Disrupt em setembro de 2011, sendo disponibilizado apenas para internet e iPhone. Apenas no terceiro semestre de 2012 o aplicativo foi lançado para Android. Mas só em maio de 2015 que a ferramenta entrou no mercado internacional, com suas versões traduzidas para o Brasil, a Alemanha e a Espanha (TRELLO, 2022).

Trata-se de uma ferramenta flexível disponível tanto em Web quanto em aplicativo para gerenciamento de trabalho onde equipes podem organizar e gerenciar projetos. Os membros podem ver várias etapas do processo e avaliar rapidamente as tarefas concluídas, além de atribuir e ver a quem foi destinado uma tarefa (SOLOMON, 2022). A estrutura do Trello é baseada nos princípios do Kanban, que consiste em uma ferramenta com colunas e cartões onde se pode acompanhar o progresso das atividades (SOLOMON, 2022).

Para o projeto a equipe utilizou o Trello em maneira conjunta as *sprints* para organizar o desenvolvimento das atividades proposta pelo professor orientador ao longo das semanas. Na Figura 10 é demonstrado a organização das tarefas, sendo divididas por semanas, onde cada *card* criado corresponde a uma tarefa pendente, contendo a data de entrega, os membros que participaram da execução, uma etiqueta indicando o andamento do afazer e a descrição com o nome do *card*.

O Trello sendo usado para administrar as atividades semanais, como demonstra a Figura 10.

FIGURA 10 - FERRAMENTA TRELLO



FONTE: OS AUTORES (2023).

3.1.4 GitHub

GitHub é um sistema de controle de versão, criado por Linus Torvalds enquanto desenvolvia do Linux, com o intuito de resolver o problema da complexidade em gerenciar várias versões de mesmo projeto. Com o GitHub, apenas uma única versão do código se torna a principal e todos da equipe podem adicionar suas alterações, tanto na versão principal quanto em *branchs*, assim centralizando o projeto e facilitando o processo de adicionar ou sincronizar funcionalidades diferentes que foram implementadas simultaneamente (POZZEBOM, 2015).

Durante o desenvolvimento do projeto do SmartCabinet, utilizamos o GitHub como repositório principal do projeto e para facilitar a manutenção do código entre a equipe.

3.1.5 Visual Studio Code

Visual Studio Code é uma IDE, simples e multiplataforma disponível para Mac, Os, Windows e Linux, totalmente configurável lançado em 2015 pela Microsoft, grátis e *open source* com seu código disponibilizado no GitHub. Ele tem suporte para uma

vasta gama de linguagens e tecnologias podendo opcionalmente conforme a necessidade serem instaladas. Ele disponibiliza várias funcionalidades que ajudam a facilitar a vida dos desenvolvedores como IntelliSense, refatoração, depuração de código, versionamento de código com o Git, automação de tarefas com Gulp, dentre outros recursos. (DEVMEDIA, 2016)

Durante o desenvolvimento do aplicativo para a gestão de remédios, o VSCode se tornou um importante aliado facilitando a escrita e leitura do código, e também trabalhando em conjunto com a ferramenta Expo Go, onde a estrutura do projeto foi montada para ser executada nessa plataforma, além de ajudar com funções rotineiras com o Git.

3.1.6 Expo Go

Trata-se de uma plataforma gratuita e de *open source*, popularizada principalmente por suportar tanto o sistema Android, quanto o IOS (EXPO GO, 2022). É uma ferramenta para desenvolvimento mobile que utiliza React Native e traz como vantagem o fácil acesso às APIs nativas do dispositivo sem possuir a necessidade de instalar qualquer dependência (FERNANDES, 2018).

O Expo Go oferece o acesso a diversos recursos como câmera, acesso bluetooth, aplicativos de forma nativa e integrada. Além de não precisar instalar o Android Studio ou o Xcode, pois a ferramenta possui suporte para as duas plataformas (FERNANDES, 2018).

Dentro do projeto o Expo Go foi utilizado em conjunto com o Visual Studio Code para o desenvolvimento do aplicativo em React Native. Um dos benefícios identificados com essa escolha foi a possibilidade executar o programa em diversas plataformas, possibilitando todos os integrantes da equipe contribuírem com o projeto.

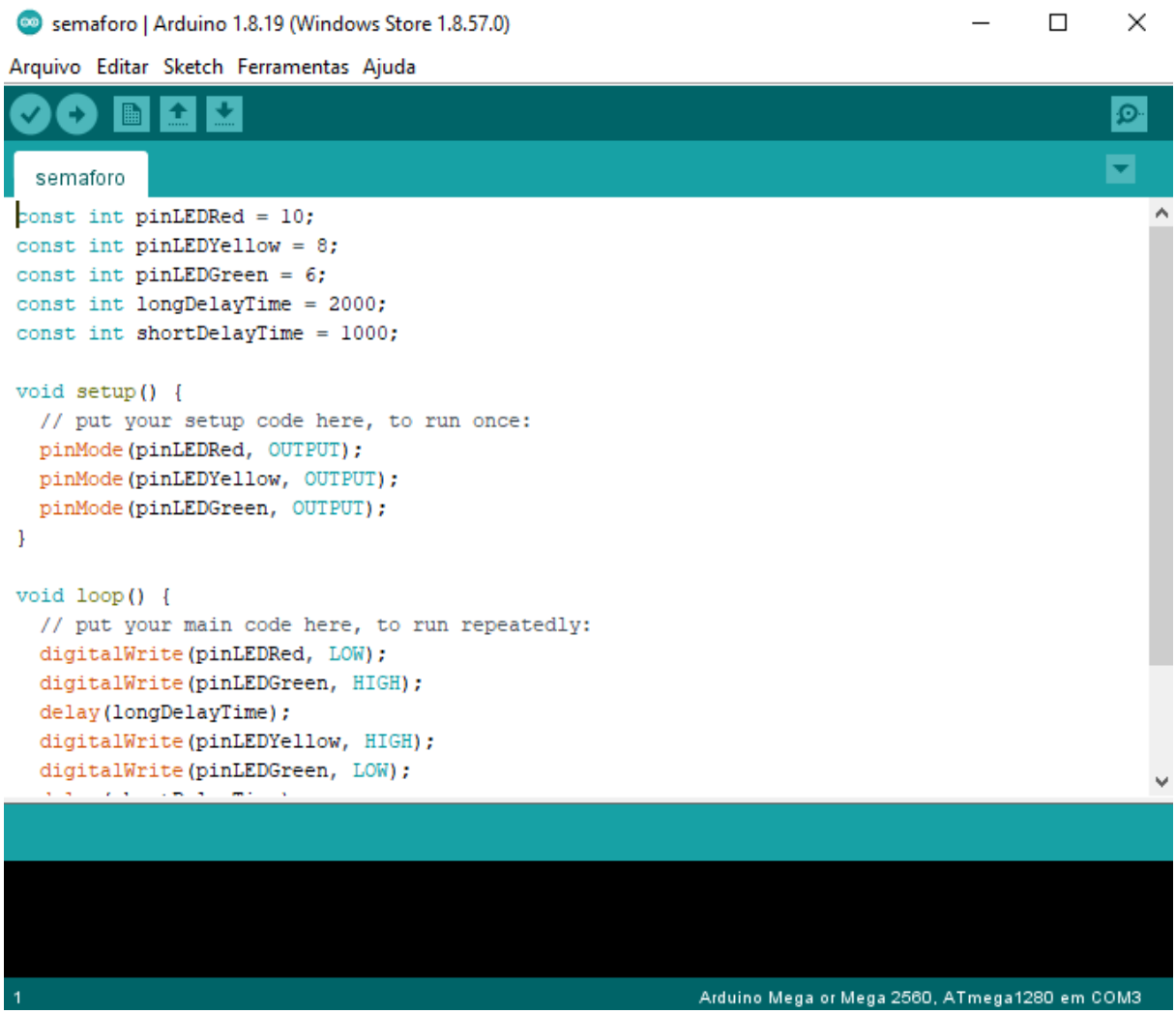
3.1.7 Arduino IDE

Arduino IDE é uma IDE para desenvolvimento embarcado para Arduino, criado em 2005 junto com o próprio Arduino por 5 pesquisadores Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino e David Mellis. Sua proposta era um dispositivo barato e simples para ajudar no desenvolvimento de estudantes e amadores. Esta IDE disponibiliza compilador e conexão com a placa, tendo

compatibilidade com todas as placas de Arduino e com ajuda de terceiros o suporte para outros equipamentos. Contamos com seu auxílio em todo o desenvolvimento para o Arduino, sendo a IDE principal para o módulo de hardware e controladores (THOMSEN, 2014).

A Figura 11 demonstra a utilização do Arduino IDE com um exemplo de código simples.

FIGURA 11 - VISÃO GERAL DA IDE ARDUINO



FONTE: OS AUTORES (2023).

3.1.8 React Native

React Native é um framework Javascript de código aberto com o padrão React, criado pelo Facebook em 2015. Sua proposta é desenvolver aplicações para

dispositivos móveis de forma nativa, dessa forma tornando o desenvolvimento mais prático e com maior desempenho, já que apenas um código funcionará tanto para Android quanto para IOS, além de proporcionar reutilização de código por ser construído com componentes (CLARK, 2020).

No projeto, o React Native foi utilizado para desenvolver todo o aplicativo, o qual auxiliará o armário de remédios ao se conectar via Wi-Fi, disponibilizando assim inúmeras funcionalidades como notificações, CRUD de medicamentos, Dashboard com as informações do armário e dos remédios, e definição de horários para cada medicação.

3.1.9 Lista de componentes

A seguir na Tabela 2 são apresentados a lista de componentes utilizados e os custos do projeto.

TABELA 2 - RESUMO DE CUSTOS SMARTCABINET

Item	Quantidade	Valor total
ARDUINO MEGA 2650	1	R$ 149,88
ETHERNET SHIELD W5100	1	R$ 152,91
BUZZER B7P-3120L	1	R$ 42,66
LED Verde	3	R$ 2,00
PROTOBOARD 400 Pontos	1	R$ 15,00
SENSOR MAGNÉTICO KY-024	3	R$ 30,00
MÓDULO RTC DS3231	1	R$ 35,00
Kit JUMPER Macho/Fêmea com 10 unidades	2	R$ 8,00
Kit JUMPER Macho/Macho com 32 unidades	1	R$ 8,00
CABO RJ-45	1	R$ 10,00
CABO USB Padrão A/b	1	R$ 10,00
Gaveteiro de mesa com 3 gavetas	1	R$ 19,80
Total		R$ 483,25

FONTE: OS AUTORES (2023).

3.2 MÉTODOS

São apresentadas na sequência as teorias utilizadas pela equipe para análise e desenvolvimento do projeto.

3.2.1 Metodologias Ágeis

O desenvolvimento ágil teve início em 2001, com o “Manifesto para o desenvolvimento ágil de software”, assinado vários programadores renomados, entre eles Kent Beck, engenheiro de software americano criador do Extreme Programming e Test Driven Development (BECK, 2001). As metodologias ágeis de desenvolvimento são constituídas por práticas e métodos que têm como objetivo tornar o desenvolvimento de software rápido, com custo controlável e melhorar a qualidade do software (FOWLER, 2005).

Para Pressman (2011, p.112), os princípios que norteiam a engenharia de software ágil visam combinar filosofia com um conjunto de princípios de desenvolvimento, onde:

> “A filosofia defende a satisfação do cliente e a entrega incremental antecipada; equipes de projeto pequenas e altamente motivadas; métodos informais; artefatos de engenharia de software mínimos; e, acima de tudo, simplicidade no desenvolvimento geral. Os princípios de desenvolvimento priorizam a entrega mais do que a análise e o projeto (embora essas atividades não sejam desencorajadas).”

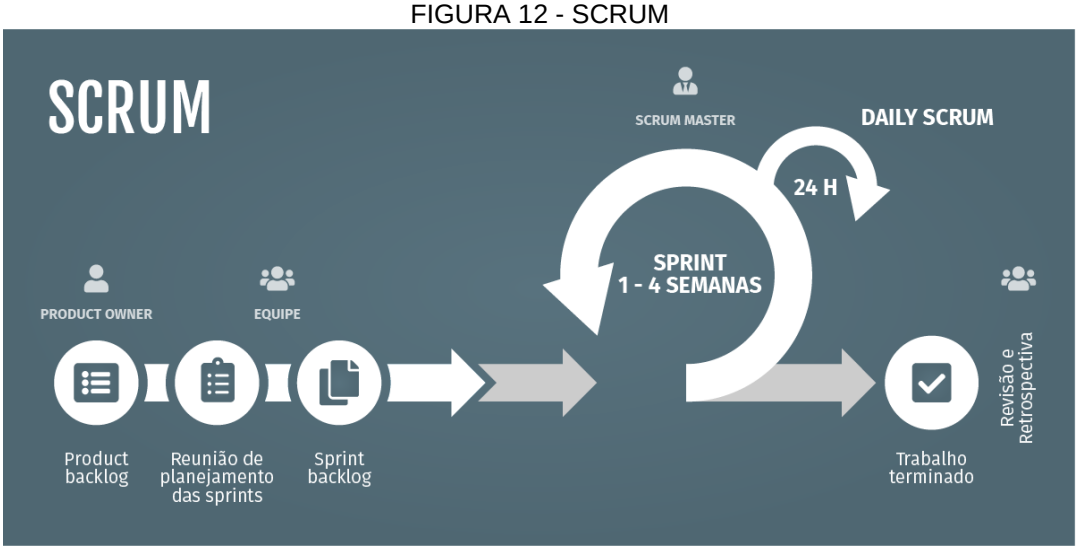
Para o desenvolvimento do projeto optamos por utilizar, de forma adaptada, as abordagens de metodologias de desenvolvimento ágil Scrum e Kanban, as quais serão abordadas nas próximas seções.

3.2.2 Scrum

A metodologia ágil Scrum segue os princípios do Manifesto Ágil e conforme Pressman (2011), é utilizada para orientar as atividades de desenvolvimento dentro de um processo que incorpora as seguintes atividades metodológicas: requisitos, análise, projeto, evolução e entrega.

O objetivo do Scrum é entregar a maior qualidade de software possível dentro de uma série de pequenos intervalos de tempo fixo, chamados *Sprints*, com duração menor que um mês (BEEDLE et al., 2000).

FIGURA 12 - SCRUM



FONTE: EMPRESÔMETRO (2020).

Na Figura 12 observa-se o funcionamento da metodologia Scrum. O processo inicia-se com a definição do *Product Backlog*, que é uma lista priorizada de requisitos para o projeto. Nesse momento, junto com o *Product Owner* (cliente do projeto), além da definição dos requisitos, são estimados custos e riscos e escolha das ferramentas de desenvolvimento, equipe e datas de entrega para os resultados (SBROCCO, 2012).

Após a definição do *Product Backlog*, ocorre a reunião de planejamento das *Sprints* junto ao *Scrum Master*, representante do cliente no projeto e responsável por garantir que a equipe siga as práticas e regras do Scrum, onde é definida a *Sprint Blacklog* (lista de funcionalidades que serão realizadas na próxima Sprint) e a divisão de tarefas entre os membros da equipe (SBROCCO, 2012).

Em seguida, dá-se início a Sprint, normalmente com duração entre 1 a 4 semanas. Durante as Sprints, ocorrem reuniões diárias entre a equipe para atualizações a sobre o andamento das atividades de cada integrante. Após o encerramento da Sprint, um incremento do produto é apresentado ao cliente e, caso seja encontrado algum problema, é adicionado ao backlog do produto. Ao fim, um novo ciclo se inicia. (SBROCCO, 2012)

Para o desenvolvimento do projeto, o Scrum foi aplicado de forma adaptada da seguinte forma:

- Sprints: Reuniões semanais junto ao professor orientador, em que a equipe apresentava as atividades realizadas durante a semana, e realizava o planejamento das tarefas para a próxima Sprint.
- Reunião de execução de atividades: Todas as semanas a equipe se reunia pelo menos uma vez, para discutir como seriam feitas as distribuições das atividades, e apresentar para o grupo o que já havia sido executado.

3.2.3 Kanban

O método Kanban, com nomenclatura de origem japonesa, que pode ser traduzida como cartão, símbolo ou painel, tem como objetivo principal oferecer visão geral de todo o projeto, e de tudo o que está sendo desenvolvido. Segundo Martins e Laugeni (2006), o Kanban preenche determinadas funções dentro do processo de produção, tais como visibilidade, onde a informação e o fluxo de material são combinados e movem-se com seus componentes, e produção, controlando a produção em seus estágios indicando o tempo, quantidade e tipo de componente a ser produzido.

Com o auxílio da ferramenta Trello, a equipe implementou o método Kanban criando um quadro com as categorias “A fazer”, “Concluído” e uma divisão para cada integrante da equipe. As atividades foram registradas em cartões, e atribuídas a sua respectiva categoria, assim auxiliando de forma visual no acompanhamento e divisão de tarefas do projeto.

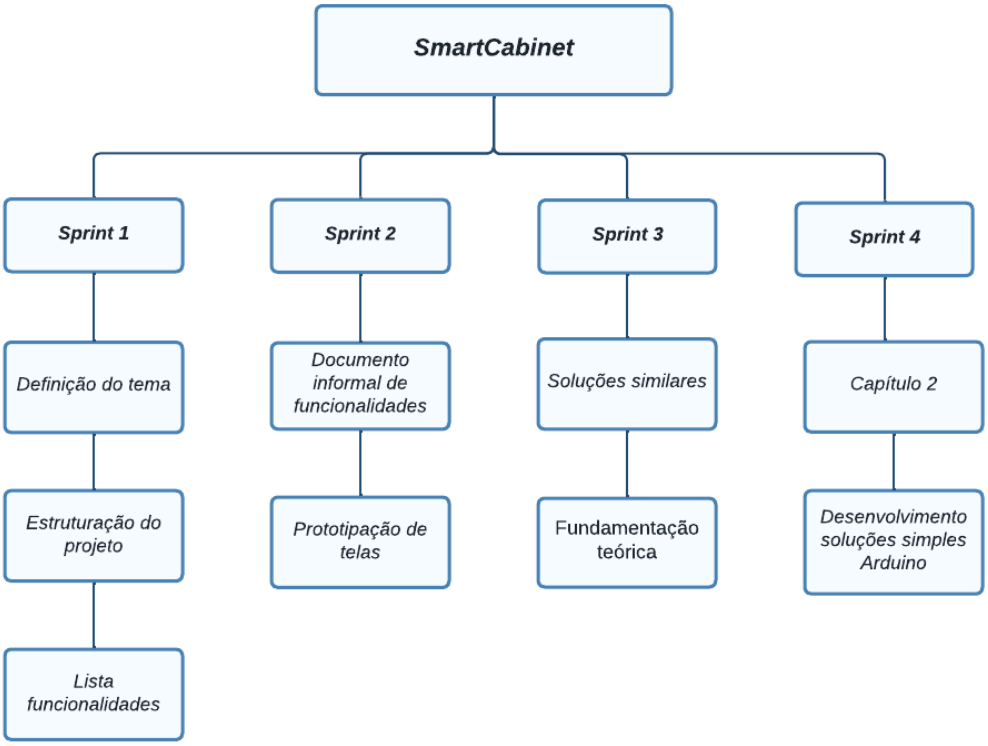
3.2.4 Plano de atividades

As atividades estabelecidas para o desenvolvimento deste projeto foram organizadas em Sprints, cada uma dessas contendo a subdivisão de entregas do trabalho, facilitando questões referentes ao gerenciamento. As atividades foram descritas graficamente através da WBS (*Work Breakdown Structure*) (Figuras 13, 14, 15 e 16) e do Gráfico de Gantt (Figuras 17, 18 e 19).

A WBS tem por objetivo subdividir um projeto em diversos componentes menores, facilmente gerenciáveis e hierarquizados, acarretando na facilidade de

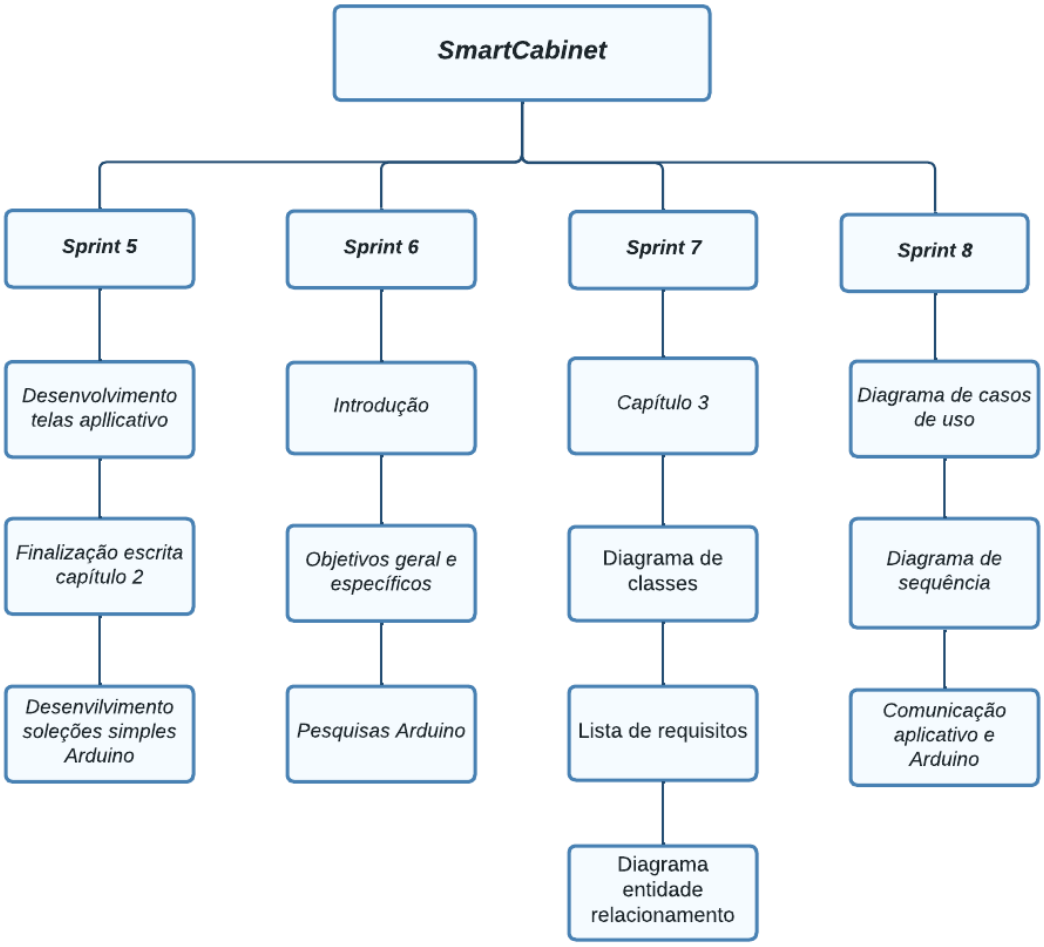
compreensão do projeto e melhor comunicação entre os participantes do mesmo (BIBLUS, 2022).

FIGURA 13 - WBS – SPRINTS 1 A 4



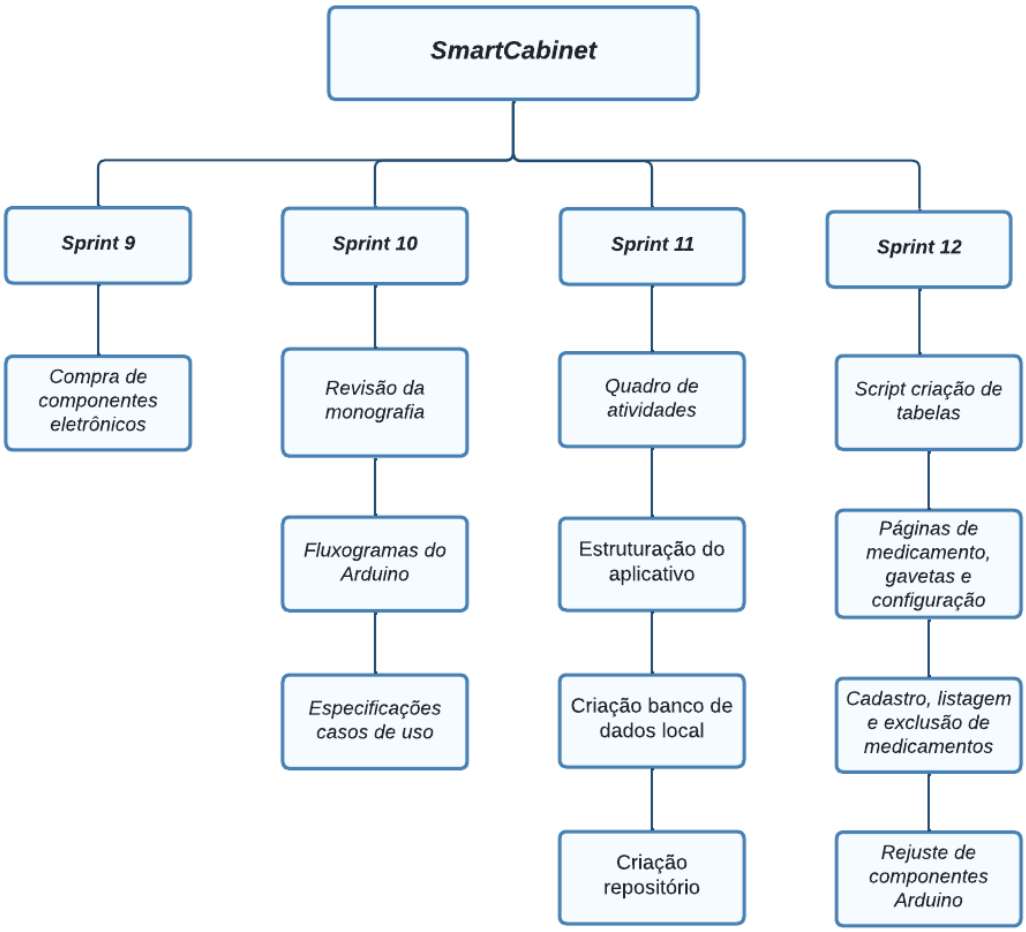
FONTE: OS AUTORES (2023).

FIGURA 14 - WBS – SPRINTS 5 A 8



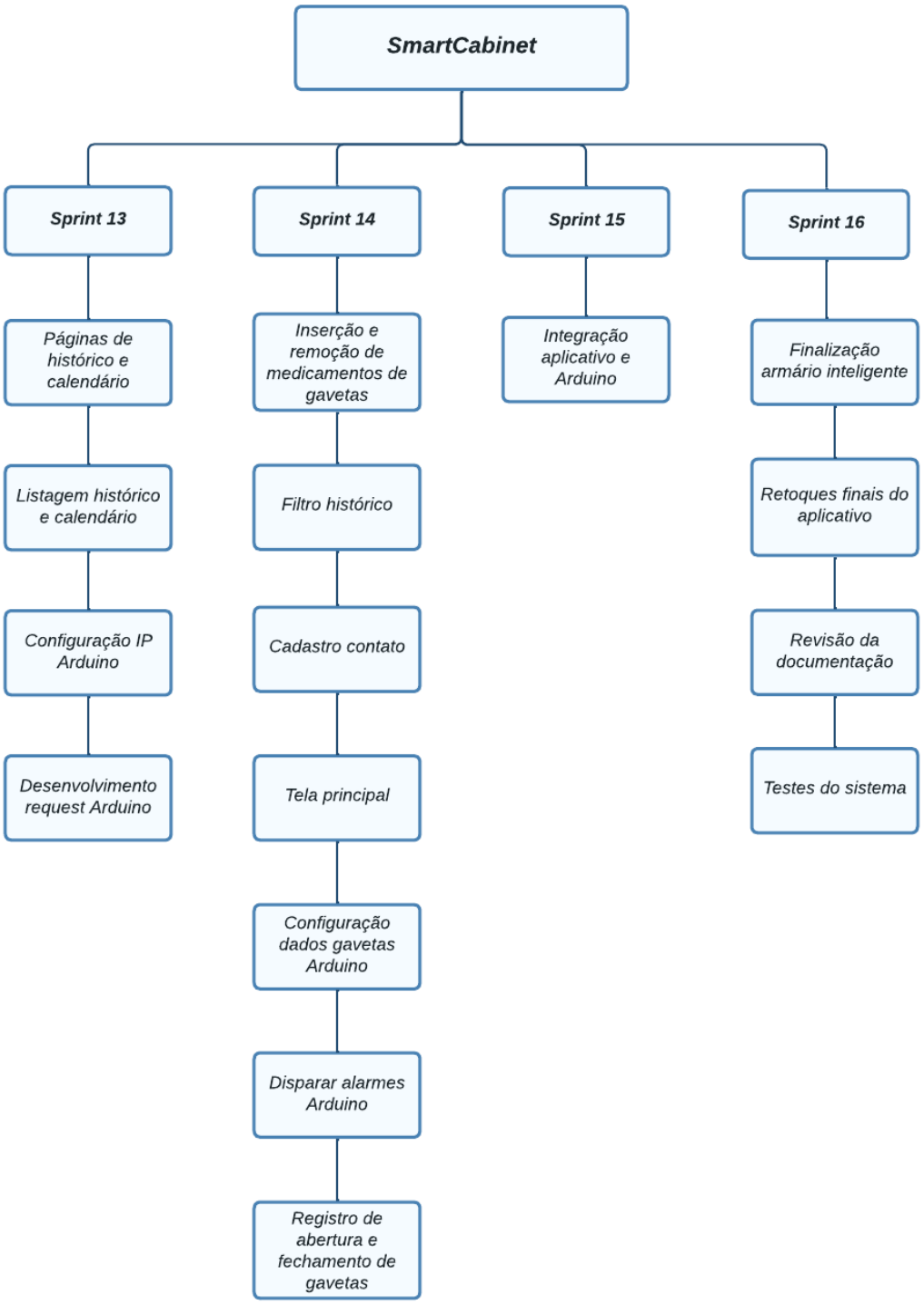
FONTE: OS AUTORES (2023).

FIGURA 15 - WBS – SPRINTS 9 A 12



FONTE: OS AUTORES (2023).

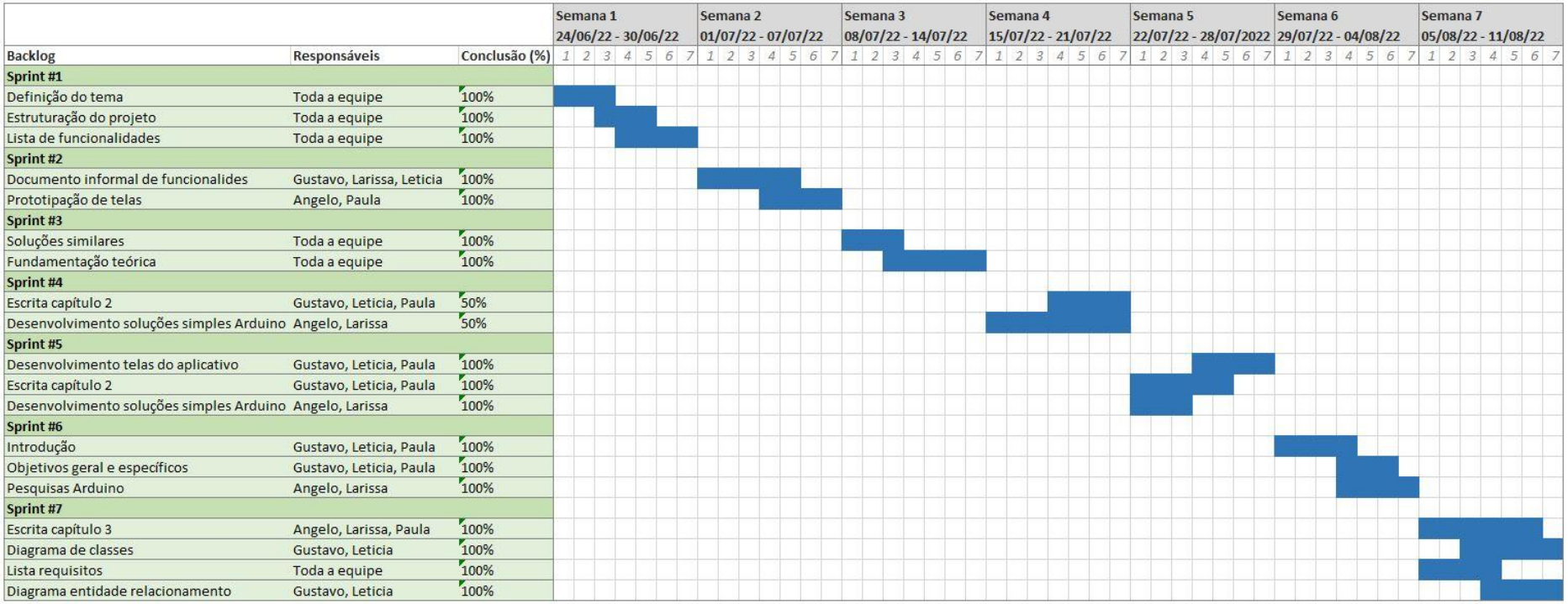
FIGURA 16 - WBS – SPRINTS 13 A 16



FONTE: OS AUTORES (2023).

O gráfico de Gantt é uma ferramenta utilizada para o gerenciamento e controle do cronograma de atividades de um projeto, sendo assim possível listar os itens necessários para o desenvolvimento do projeto, o responsável pela atividade em questão e a estimativa de tempo para execução (ESPINHA, 2021). Nele foram descritas as atividades de cada Sprint, seus respectivos desenvolvedores, percentual de conclusão, e período de duração (Figuras 17, 18 e 19).

FIGURA 17 - GRÁFICO DE GANTT - SPRINTS 1 A 7

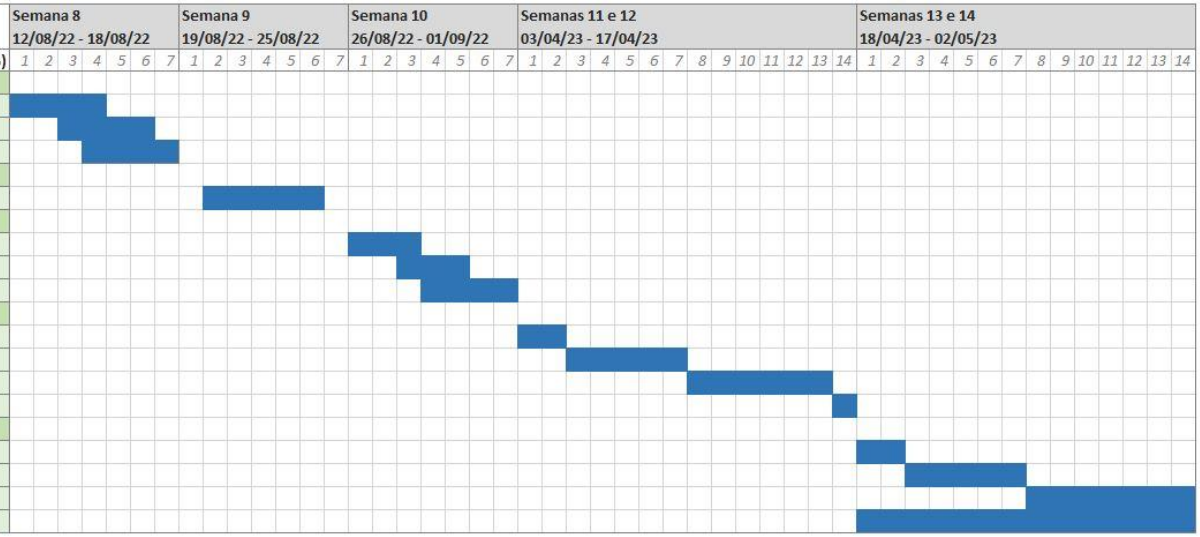


Backlog	Responsáveis	Conclusão (%)
Sprint #1		
Definição do tema	Toda a equipe	100%
Estruturação do projeto	Toda a equipe	100%
Lista de funcionalidades	Toda a equipe	100%
Sprint #2		
Documento informal de funcionalides	Gustavo, Larissa, Leticia	100%
Prototipação de telas	Angelo, Paula	100%
Sprint #3		
Soluções similares	Toda a equipe	100%
Fundamentação teórica	Toda a equipe	100%
Sprint #4		
Escrita capítulo 2	Gustavo, Leticia, Paula	50%
Desenvolvimento soluções simples Arduino	Angelo, Larissa	50%
Sprint #5		
Desenvolvimento telas do aplicativo	Gustavo, Leticia, Paula	100%
Escrita capítulo 2	Gustavo, Leticia, Paula	100%
Desenvolvimento soluções simples Arduino	Angelo, Larissa	100%
Sprint #6		
Introdução	Gustavo, Leticia, Paula	100%
Objetivos geral e específicos	Gustavo, Leticia, Paula	100%
Pesquisas Arduino	Angelo, Larissa	100%
Sprint #7		
Escrita capítulo 3	Angelo, Larissa, Paula	100%
Diagrama de classes	Gustavo, Leticia	100%
Lista requisitos	Toda a equipe	100%
Diagrama entidade relacionamento	Gustavo, Leticia	100%

FONTE: OS AUTORES (2023).

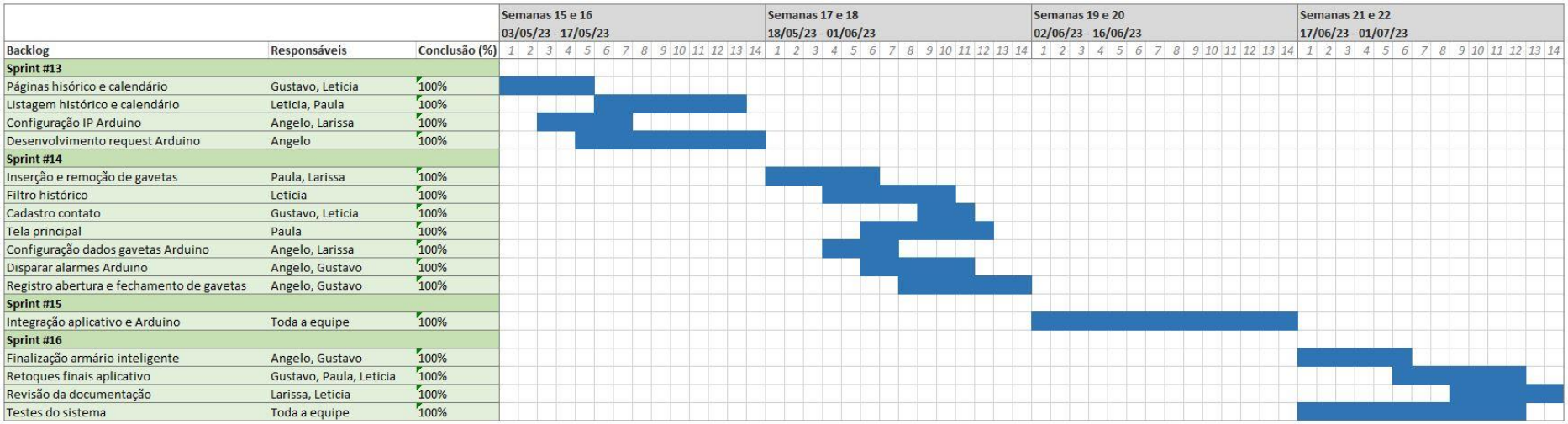
FIGURA 18 - GRÁFICO DE GANTT - SPRINTS 8 A 12

Backlog	Responsáveis	Conclusão (%)
Sprint #8		
Diagrama de casos de uso	Angelo, Larissa, Paula	100%
Diagrama de sequência	Gustavo, Leticia	100%
Comunicação aplicativo e Arduino	Toda a equipe	100%
Sprint #9		
Compra de componentes eletrônicos	Toda a equipe	100%
Sprint #10		
Revisão monografia	Larissa	100%
Especificações casos de uso	Leticia, Paula	100%
Fluxogramas Arduino	Angelo, Gustavo	100%
Sprint #11		
Quadro de atividades	Toda a equipe	100%
Estruturação do aplicativo	Toda a equipe	100%
Criação banco de dados local	Toda a equipe	100%
Criação repositório	Toda a equipe	100%
Sprint #12		
Script criação de tabelas	Gustavo, Leticia, Paula	100%
Páginas de medicamento, gavetas e configuração	Gustavo, Leticia, Paula	100%
Cadastro, listagem e exclusão de medicamentos	Gustavo, Leticia, Paula	100%
Reajuste componentes Arduino	Angelo, Larissa	100%



FONTE: OS AUTORES (2023).

FIGURA 19 - GRÁFICO DE GANTT – SPRINTS 13 A 16



Backlog	Responsáveis	Conclusão (%)
Sprint #13		
Páginas hisórico e calendário	Gustavo, Leticia	100%
Listagem histórico e calendário	Leticia, Paula	100%
Configuração IP Arduino	Angelo, Larissa	100%
Desenvolvimento request Arduino	Angelo	100%
Sprint #14		
Inserção e remoção de gavetas	Paula, Larissa	100%
Filtro histórico	Leticia	100%
Cadastro contato	Gustavo, Leticia	100%
Tela principal	Paula	100%
Configuração dados gavetas Arduino	Angelo, Larissa	100%
Disparar alarmes Arduino	Angelo, Gustavo	100%
Registro abertura e fechamento de gavetas	Angelo, Gustavo	100%
Sprint #15		
Integração aplicativo e Arduino	Toda a equipe	100%
Sprint #16		
Finalização armário inteligente	Angelo, Gustavo	100%
Retoques finais aplicativo	Gustavo, Paula, Leticia	100%
Revisão da documentação	Larissa, Leticia	100%
Testes do sistema	Toda a equipe	100%

FONTE: OS AUTORES (2023).

3.2.5 Responsabilidades

Após o planejamento inicial do projeto, a realização de pesquisas a respeito das tecnologias a serem utilizadas, e por sugestão do orientador, a divisão de tarefas do projeto realizou-se da seguinte forma: Escrita da monografia, desenvolvimento do aplicativo, e criação de soluções com o Arduino.

Apesar de cada membro focar no grupo de atividade proposto, todos maneira recorrente auxiliavam outros membros da equipe em seus seguimentos.

A seguir, observa-se a Tabela 1 representando as principais responsabilidades de cada um dos integrantes.

TABELA 3 - QUADRO DE RESPONSABILIDADE

RESPONSABILIDADES	
Integrantes	Atividades
Angelo Trombim	Prototipagem das telas; Escrita da monografia; Gráfico de Gantt; Gráfico WBS; Desenvolvimento de soluções simples de Arduino; Integração do aplicativo com o Arduino; Construção do Armário inteligente.
Gustavo Brito	Escrita da monografia; Lista de requisitos; Diagrama de Caso de Uso; Diagrama de Classes; Fluxograma; Integração do aplicativo com o Arduino; Desenvolvimento do aplicativo.
Larissa Neves	Escrita da monografia; Desenvolvimento de soluções simples de Arduino; Desenvolvimento do aplicativo.
Leticia Paumer	Escrita da monografia; Desenvolvimento do aplicativo; Diagrama de Caso de Uso; Diagrama de Classes; Diagrama de Sequência; Diagrama Entidade Relacionamento;

	Fluxograma.
Paula Sucla	Prototipagem das telas; Escrita da monografia; Desenvolvimento do aplicativo; Desenvolvimento de soluções simples de Arduino.

FONTE: OS AUTORES (2023).

3.3 DESENVOLVIMENTO DO PROJETO

Para a execução do projeto, foram definidas 16 *sprints,* sendo as 10 primeiras com duração de uma semana, e as 6 últimas com duração de duas semanas. Também foram elaborados os requisitos (Apêndice A), os diagramas de casos de uso e suas especificações (Apêndices B e C, respectivamente), diagrama de classes (Apêndice D), diagramas de sequência (Apêndice D), diagrama entidade relacionamento (Apêndice F), fluxograma do Arduino (Apêndice G) e Especificação da API do Arduino (Apêndice H).

Nas próximas sessões serão descritas, de forma detalhada, cada uma das *sprints* realizadas ao decorrer do projeto.

3.3.1 Sprint 1

Na primeira *sprint* foram debatidos os primeiros passos do projeto, como a definição do tema, funcionalidades e estrutura. Para isso, foram realizadas a leitura de diversos outros trabalhos de conclusão de curso e artigos científicos visando a compreender de modo geral a construção do documento.

Em seguida foi realizada uma reunião entre a equipe para definir a lista de funcionalidades do projeto, tanto do armário como as do aplicativo.

3.3.2 Sprint 2

Na segunda *sprint* foi definido junto ao professor orientador um documento informal de funcionalidades, especificando de forma detalhada cada funcionalidade do aplicativo e do armário. Também foi realizada a prototipação das telas do aplicativo utilizando a ferramenta INVISION.

3.3.3 Sprint 3

Na terceira *sprint*, foi realizado um estudo de soluções similares ao tema que foi proposto neste trabalho que já estão inseridos no mercado atualmente. Além disso, foram feitas diversas pesquisas em artigos, trabalhos científicos e sites, com foco no tema “Administração incorreta de medicamentos” visando o embasamento teórico do problema apresentado por este trabalho.

3.3.4 Sprint 4

Durante a quarta *sprint*, foi dado início a construção do documento, começando pela escrita do Capítulo 2 - Fundamentação Teórica, incluindo o conceito e embasamento do problema, tecnologias e trabalhos relacionados.

Em paralelo, iniciou-se o desenvolvimento de soluções simples utilizando Arduino para melhor compreensão do funcionamento desta plataforma.

3.3.5 Sprint 5

Durante a semana da quinta *sprint* foi finalizada a escrita do Capítulo 2 - Fundamentação Teórica, e o desenvolvimento do aplicativo foi iniciado com o desenvolvimento das telas de gaveta e cadastro de um medicamento. Também foi dado continuidade ao desenvolvimento de soluções simples com Arduino.

3.3.6 Sprint 6

Na sexta *sprint* ocorreu a documentação dos seguintes tópicos: introdução, justificativa, objetivo geral, objetivos específicos e estrutura do documento. Além disso, foi dado início a escrita do Capítulo 3 - Especificação.

Por fim, iniciou-se pesquisas relacionadas a plataforma do Arduino e como estabelecer comunicação entre o Arduino e o aplicativo utilizando conexão Wi-Fi.

3.3.7 Sprint 7

A sétima *sprint* foi focada na finalização do capítulo 3 da documentação. Nessa semana também foi trabalhado na modelagem do sistema, iniciando pelo diagrama

de classes (APÊNDICE D), lista de requisitos (APÊNDICE A) e diagrama entidade relacionamento (APÊNDICE F).

A realização da comunicação entre o Arduino e o aplicativo precisou ser adiada devido a falta do equipamento necessário.

3.3.8 Sprint 8

A oitava *sprint* foi focada na realização dos ajustes necessários na documentação e na realização dos diagramas de casos de uso (APÊNDICE B) e diagramas de sequência (APÊNDICE E).

Após a obtenção do equipamento necessário, foi dado continuidade na comunicação entre o Arduino e o aplicativo. Foi criado um componente na tela, que ao ser acionado produzia uma resposta no Arduino.

3.3.9 Sprint 9

Na nona *sprint* ocorreu a compra, junto ao professor orientador, dos componentes do Arduino necessários para a construção do armário inteligente.

Após a aquisição dos produtos, houve uma breve discussão sobre as alterações necessárias nos Capítulos 1 e 2 do documento.

3.3.10 Sprint 10

Na décima *sprint*, houve a revisão e correção da monografia. As alterações realizadas focaram-se nos dois primeiros capítulos, pois havia dúvidas a respeito dos tópicos que não tinham sido discutidos na *sprint* anterior. Além disso, foram desenvolvidos os fluxogramas do Arduino (APÊNDICE G) e as especificações dos casos de uso (APÊNDICE C).

3.3.11 Sprint 11

Na décima primeira *sprint*, iniciou-se uma nova fase do projeto, onde a equipe passou a focar no desenvolvimento e programação do aplicativo e do Arduino. Para esta nova etapa do projeto, foi criado um quadro de atividades na ferramenta Trello, auxiliando a equipe com a divisão de tarefas e estabelecimento de prazos.

Durante a *sprint* também foi feita a estruturação do aplicativo e criação de um banco de dados local via SQL Lite.

Para a equipe poder trabalhar em conjunto no desenvolvimento do aplicativo, foi criado também um repositório no GitHub.

3.3.12 Sprint 12

Durante a décima segunda *sprint*, em relação ao aplicativo, foram desenvolvidas as seguintes atividades: desenvolvimento dos scripts para criação de tabelas no banco de dados; criação das páginas de medicamentos, gavetas e configuração; desenvolvimento das funcionalidades de cadastro, listagem e exclusão de medicamentos.

Em relação ao Arduino, foi necessário fazer um reajuste de componentes para dar continuidade no desenvolvimento. Nessa *sprint* também foi realizada a compra do armário com quatro gavetas para o projeto.

3.3.13 Sprint 13

Na décima terceira *sprint* foram desenvolvidas as telas de histórico e calendário do aplicativo. Também foram implementadas funcionalidades de listagem dessas mesmas telas.

No desenvolvimento no Arduino, trabalhou-se na configuração do IP da rede local para comunicação do Arduino com o aplicativo e no desenvolvimento da *request* que escuta as solicitações do aplicativo.

3.3.14 Sprint 14

Na décima quarta *sprint*, no desenvolvimento do aplicativo, foram realizadas seguintes atividades: funcionalidades de inserção e remoção de medicamentos das gavetas; criação da tela de filtro do histórico e implementação dos códigos de filtragem; cadastro de contato; desenvolvimento da tela principal.

No desenvolvimento no Arduino, foi criado o código para configurar dados da gaveta e disparar os alarmes nos horários agendados. Também foi desenvolvido o código para registro de abertura e fechamento de gavetas.

3.3.15 Sprint 15

Durante a semana da décima quinta *sprint*, o maior foco da equipe foi na integração do aplicativo com o Arduino. Com o desenvolvimento individual de ambas as partes do projeto concluído, trabalhamos com a comunicação entre o Arduino e a aplicativo, manipulando a troca de informações entre eles. Durante a sprint também foi desenvolvida a especificação da API do Arduino (Apêndice H).

3.3.16 Sprint 16

Na décima sexta *sprint*, fase final do projeto, foram realizadas as seguintes atividades: finalização da construção do armário inteligente, integrando os componentes eletrônicos com o armário; retoques finais no aplicativo, com foco em correções de layout e melhorias gerais; revisão e correção da documentação do projeto; realização de testes do funcionamento do sistema.

4 APRESENTAÇÃO DO SISTEMA

Neste capítulo será apresentada, de forma completa e detalhada, toda a estruturação e funcionalidades do sistema SmartCabinet. As próximas sessões descrevem a arquitetura geral do sistema, e demostram composição do armário e a utilização do aplicativo.

4.1 ARQUITETURA DO SISTEMA

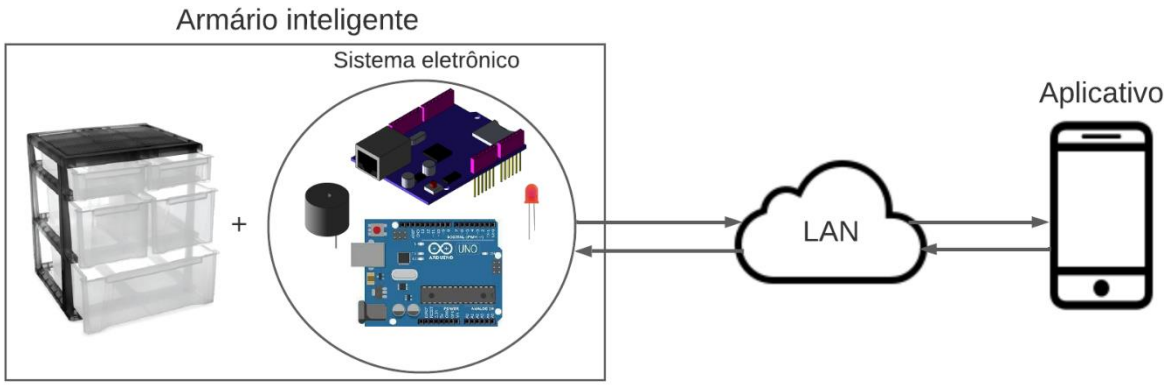
O projeto foi dividido em duas partes. A primeira parte consiste em um armário físico inteligente, construído a partir da junção de um pequeno armário com um sistema eletrônico composto por: Arduino Mega, Protoboard, Shield Ethernet, sensores magnéticos, LEDs, Buzzer e um Módulo RTC. Na outra parte do projeto, temos uma aplicação para dispositivos móveis, responsável pelo gerenciamento de medicamentos e andamento do tratamento do usuário. Para o funcionamento desse sistema, a troca de informações entre o aplicativo e o armário inteligente é essencial. Na Figura 20, podemos visualizar a estrutura geral e o fluxo de comunicação do projeto SmartCabinet.

Para a construção da parte elétrica armário, foi utilizada uma Protoboard para conexão dos circuitos eletrônicos, e um Shield Ethernet na comunicação do Arduino com o aplicativo via rede local. No reconhecimento de abertura ou fechamento de gavetas, foram utilizados sensores magnéticos. Por fim, foram instaladas luzes de LED e um Buzzer para notificar o usuário de forma visual e sonora nos horários de cada gaveta, e um Módulo RTC para manter informações armazenadas ao desligar a fonte de energia.

O Arduino teve papel fundamental neste projeto, sendo responsável por armazenar informações de medicamentos enviadas pelo aplicativo, possibilitando disparar alertas ao usuário nos horários cadastrados para cada medicamento, a partir do desenvolvimento de códigos de agendamento na linguagem de programação C++. Além disso, o Arduino também foi encarregado de registrar e armazenar os dados de abertura e fechamento de gavetas, e posteriormente enviar essas informações para o aplicativo, disponibilizando ao usuário dados do andamento de seu tratamento.

O aplicativo, desenvolvido a partir do *framework* React Native e emulado e testado junto a ferramenta Expo Go, é responsável por fornecer ao usuário uma forma de gerenciar seus medicamentos e trocar informações com o Arduino quanto o seu tratamento. A aplicação permite o armazenamento local de dados e geração de relatórios do processo do paciente.

FIGURA 20 - REPRESENTAÇÃO DO PROJETO



FONTE: OS AUTORES (2023).

4.2 ARDUINO

Na Figura 21, podemos visualizar o mapa eletrônico com todos os componentes utilizados no armário inteligente. Especificando cada um dos componentes enumerados no mapa, temos:

1. Protoboard: Uma ferramenta utilizada para simular uma placa-mãe, possibilitando conectar diversos componentes eletrônicos.

2. Modulo RTC (*Real Time Clock):* É um módulo independente utilizado para guardar, manipular e retornar informações sobre data e hora. Capaz de se manter ativo sem uma fonte de energia por mais de 6 meses, pois possui bateria própria. No projeto, o módulo auxilia nos cálculos de intervalos e verificação de ocorrência de medicamentos agendados.

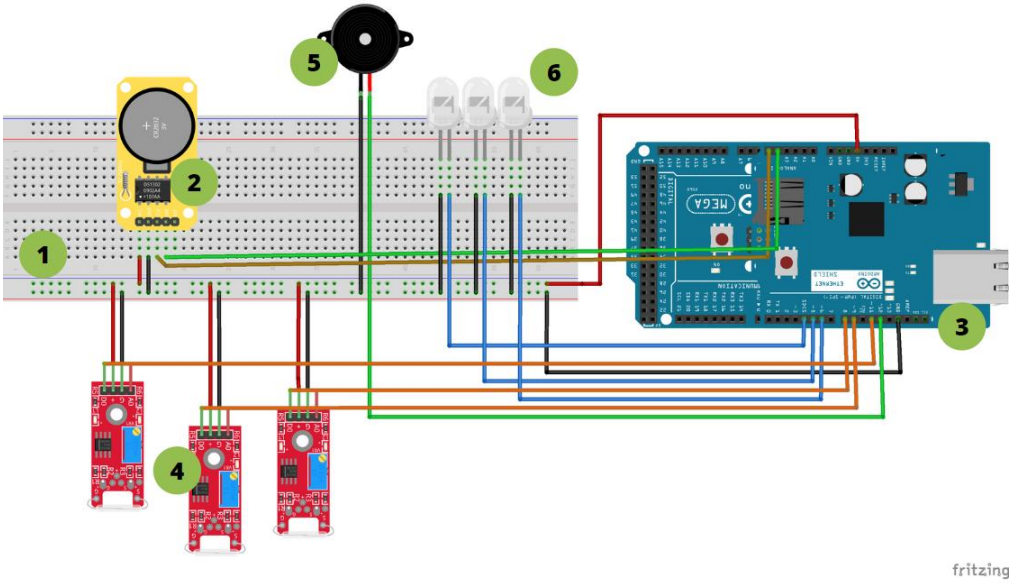
3. Arduino Mega 2650 + Shield ethernet: O Arduino Mega 2650 é uma evolução robusta do Arduino Uno, que traz maior memória RAM, memória interna e portas. Junto ao Shield ethernet, o Arduino consegue conectar-se ao cabo RJ-45 e acessar a rede local ou internet.

4. Sensores magnéticos: Sensores capazes de identificar variações do campo magnético, possibilitando o registro de abertura e/ou fechamento das gavetas.

5. Buzzer: Dispositivo eletrônico de sinalização de áudio.

6. Led: Dispositivo eletrônico para emissão de luz.

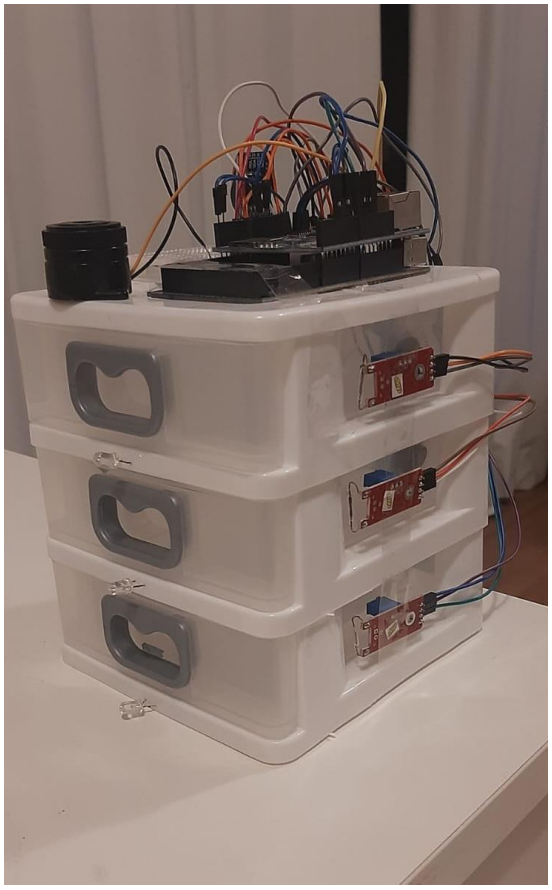
FIGURA 21 - MAPA ELETRÔNICO



FONTE: OS AUTORES (2023).

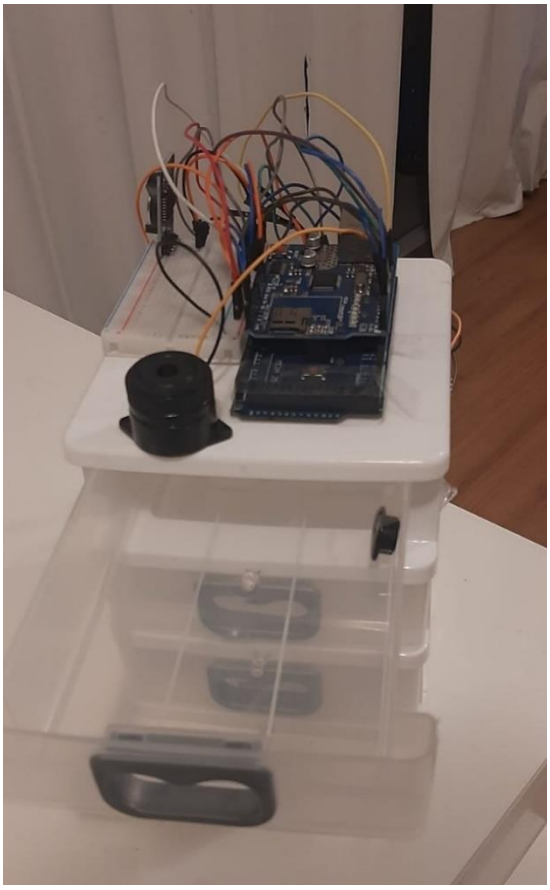
Na Figuras 22 e 23, temos o armário inteligente finalizado e completamente funcional. Ele dispõe de três gavetas, as quais podem ser utilizadas para diferentes medicamentos.

FIGURA 22 - ARMÁRIO INTELIGENTE FUNCIONAL



FONTE: OS AUTORES (2023).

FIGURA 23 - ARMÁRIO FUNCIONAL GAVETA



FONTE: OS AUTORES (2023).

4.3 APLICATIVO

A tela inicial do aplicativo móvel (Figura 24) contém um dashboard, onde são apresentadas ao usuário diversas informações relacionadas ao seu tratamento, sendo elas: horário do próximo medicamento, sequência de dias de remédios tomados no horário correto, quantidade de medicamentos não tomados, nome do último remédio ingerido e um gráfico contendo informações quanto a quantidade de medicamentos em cada gaveta.

FIGURA 24 - TELA DASHBOARD

FONTE: OS AUTORES (2023).

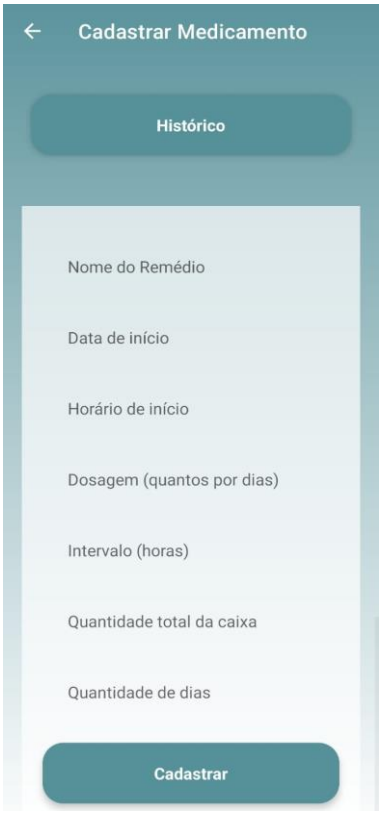
A partir da tela inicial, o usuário pode acessar pelo menu inferior as páginas de histórico, medicamento, gavetas, calendário e configurações.

Na tela de medicamentos, é possível visualizar todos os já cadastrados (Figura 25), adicionar um novo medicamento (Figura 26), modificar informações de um medicamento já cadastrado ou excluir um medicamento (Figura 27).

FIGURA 25 - TELA MEDICAMENTOS

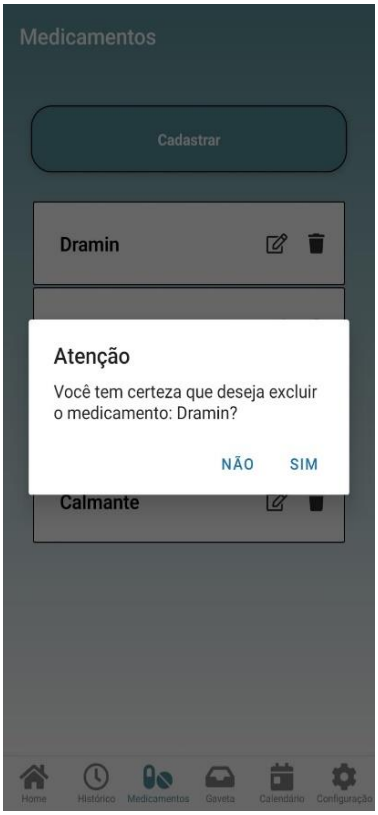
FONTE: OS AUTORES (2023).

FIGURA 26 - TELA CADASTRO MEDICAMENTO



FONTE: OS AUTORES (2023).

FIGURA 27 - TELA REMOÇÃO MEDICAMENTO



FONTE: OS AUTORES (2023).

Ao acessar a página de gavetas, o usuário é redirecionado a uma lista de gavetas, as quais são correspondentes as gavetas físicas do armário (Figura 28). Para adicionar um medicamento a gaveta, deve-se selecionar uma gaveta vazia e escolher um remédio já cadastrado no sistema (Figura 29). Para a remoção de um medicamento da gaveta, o usuário deve selecionar uma gaveta ocupada e clicar na opção “limpar gaveta” (Figura 30).

FIGURA 28 - TELA INICAL GAVETAS

FONTE: OS AUTORES (2023).

FIGURA 29 - TELA ADICIONAR GAVETA

FONTE: OS AUTORES (2023).

FIGURA 30 - TELA REMOVER GAVETA

FONTE: OS AUTORES (2023).

Na tela inicial de histórico, é gerada uma lista dos últimos medicamentos ingeridos pelo paciente, contendo dados do nome do remédio, horário agendado e horário de abertura da gaveta (Figura 31). Na mesma página, ao clicar no botão de “filtrar”, o usuário é redirecionado para o formulário de filtro (Figura 32), onde ele pode aplicar os filtros desejados e visualizar o histórico filtrado (Figura 33).

FIGURA 31 - TELA HISTÓRICO

FONTE: OS AUTORES (2023).

FIGURA 32 - TELA FILTRO

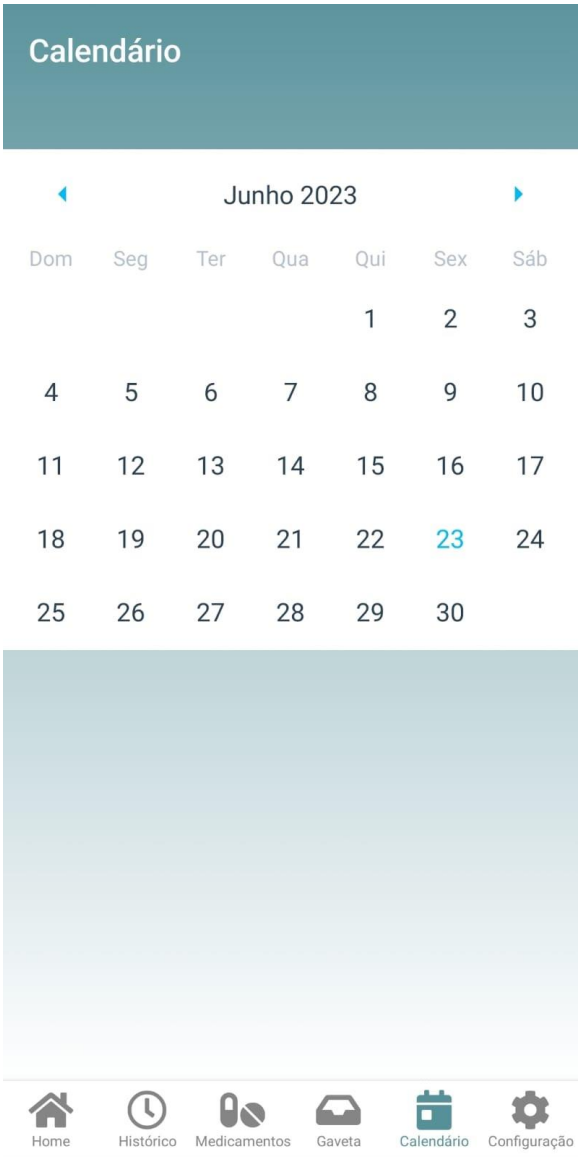
FONTE: OS AUTORES (2023).

FIGURA 33 - TELA HISTÓRICO FILTRADO

FONTE: OS AUTORES (2023).

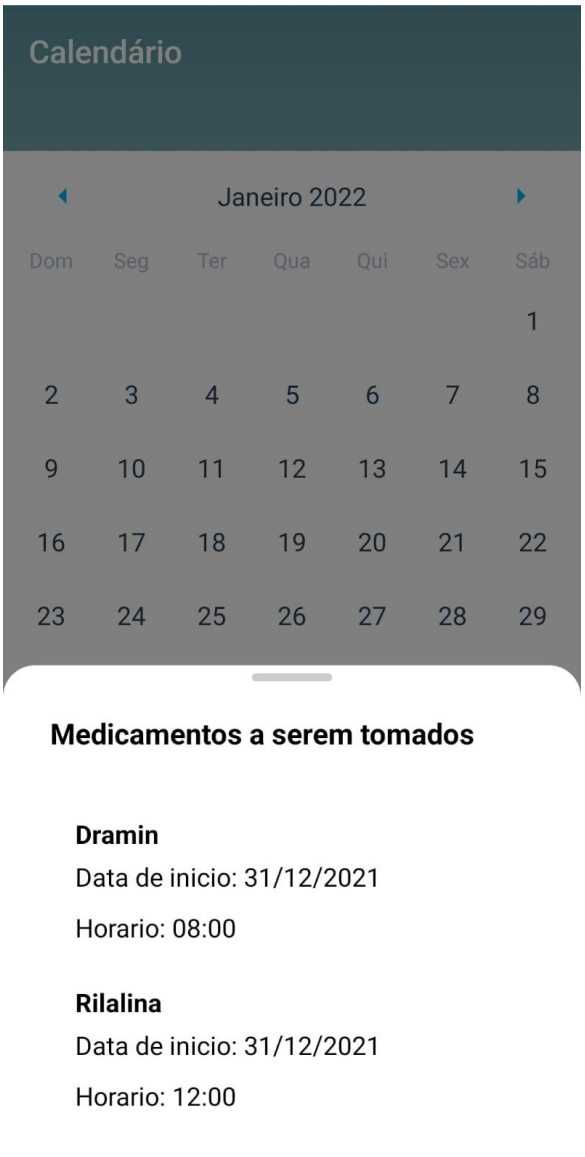
Na tela de calendário (Figura 34), o sistema disponibiliza um calendário onde o usuário pode selecionar um determinado dia para visualizar os medicamentos agendados para aquela data (Figura 35).

FIGURA 34 - TELA CALENDÁRIO



FONTE: OS AUTORES (2023).

FIGURA 35 - TELA DIA SELECIONADO



FONTE: OS AUTORES (2023).

Ao acessar a tela de configuração (Figura 36), clicando na opção “Contatos”, o usuário pode cadastrar o número de telefone de um contato, ou alterar/remover um contato já cadastrado. Na opção “Enviar relatório” o usuário pode mandar informações sobre o seu tratamento para o número do contato cadastrado via Whatsapp.

FIGURA 36 - TELA CONFIGURAÇÕES

FONTE: OS AUTORES (2023).

5 CONSIDERAÇÕES FINAIS

Ao longo do desenvolvimento do projeto, notamos que a dificuldade de administração de medicamentos não se restringe a um grupo específico, mas sim, que atinge a população global como um todo. Os estudos mostraram que muitos dos eventos adversos ao longo de um tratamento médico são evitáveis ao seguirmos de maneira correta as prescrições médicas. Além do agravamento de um quadro clínico e possíveis sequelas, a gestão errônea dos fármacos pode contribuir para gastos excedentes e ineficiência do remédio utilizado. A má conduta também afeta órgãos públicos, gerando consequências financeiras ao governo, e as unidades de saúde, onde a verba e a prioridade em atendimentos são destinadas a pacientes que não precisariam recorrer à uma unidade médica.

A partir dessas observações, o desenvolvimento do projeto é conduzido com o intuito de auxiliar a população na administração correta de medicamentos, buscando concluir um tratamento médico com êxito. Além de mirrar possíveis adversidades em decorrência do mau uso desses remédios, como complicações clínicas, encarecimento do tratamento, e sequelas ao paciente.

Na primeira fase do projeto, a equipe teve como objeto a pesquisa, e estudos de casos, tentando compreender os principais obstáculos dos pacientes. A análise desses requisitos mostrou-se essencial para a determinação das aplicações implementadas.

Assim, planejou-se a construção de um sistema para *Smartphones*, conectado a um *IoT* em forma de armário inteligente, que tem como principal objetivo notificar o usuário a respeito dos horários que se deve ingerir determinado medicamento, além de fornecer informações para maior clareza do tratamento. Para isso, o aplicativo propõe agendamento de medicamentos, relatório de dados, avisos sonoros e visuais.

Durante a elaboração do projeto e no decorrer das Sprints, houve uma necessidade por parte dos membros da equipe de aperfeiçoarem suas habilidades em desenvolvimento de aplicações para dispositivos móveis, além de compreender fundamentações da eletrônica, funcionamento de hardware e a integração de *IoTs*.

Como resultado, foi desenvolvido um sistema conectado via rede local a um Arduino acoplado a um armário, onde ao cadastrar uma medicação e alocá-lo em uma gaveta disponível, o programa calcula os horários em que o equipamento deve notificar o usuário através de aviso visual (Led) e sonoro (Buzzer). O aplicativo conta

ainda com a visualização de informações relevantes para o controle do tratamento, relatório com data e hora da ingestão de cada medicamento, facilidade ao cadastrar remédios de uso contínuo, além de um calendário para verificar datas posteriores ou futuras de um determinado tratamento.

Para obtenção do produto desejado, foi utilizada a linguagem React Native, que trata-se de uma biblioteca multiplataforma, junto a ferramenta Expo Go que abstrai parte da complexidade da configuração de um ambiente e permite de maneira fácil e rápida emular e testar um aplicativo em construção. Já no Arduino, a implementação da lógica de seu funcionamento foi construída através da linguagem C++. A integração entre o dispositivo móvel e o Arduino são de suma importância para o funcionamento total do projeto, visto que, as informações utilizadas e funcionalidades aplicadas então todas relacionadas. Os dados recebidos pelo aplicativo através de requisições, são manipulados para agendar de maneira correta os horários do armário inteligente. Além disso, o hardware retorna o registro de abertura das gavetas do equipamento, possibilitando que o sistema catalogue a consistência do tratamento.

Contudo, surgiram algumas limitações no decorrer do projeto, devido a constituição do Arduino, não foi possível manipular grandes quantidades de dados em seu sistema. Houve também dificuldades durante o desenvolvimento tanto do aplicativo quanto do código do hardware, pois tratava-se de linguagens e plataformas que os membros não tinham conhecimento prévio, o que acarretou na evolução mais lenta do projeto. Além do mais, nenhum dos integrantes possuía conhecimento em eletrônica, sendo necessário estudos desde os conceitos mais básicos.

5.1 RECOMENDAÇÕES PARA TRABALHOS FUTUROS

As recomendações para trabalhos futuros contam com a implementação de um dispositivo de armazenamento como um SD no Arduino, ou o uso de sistema de cloud, onde será possível guardar grandes volumes de dados, mantendo os registros mais completos e consistentes. Também, para o hardware, a alteração do *Shield ethernet* para uma placa com Wi-Fi pode auxiliar na disposição do armário no ambiente do usuário.

Já no aplicativo, notificações através do Whatsapp e das funções nativas dos dispositivos móveis, mostram-se favoráveis para contribuir com a proposta do projeto.

Na estrutura do armário, é interessante desenvolver um método para que o paciente tenha acesso apenas a dose do horário indicado, e não de todos os comprimidos, evitando o consumo errôneo da medicação e promovendo maior precisão da quantidade de remédios de cada gaveta.

REFERÊNCIAS

ANDERSON, J. G. et al. **Evaluating the capability of information technology to prevent adverse drug events: a computer simulation approach**. J. Am. Med. Inform Assoc., 2002. Disponível em: https://pubmed.ncbi.nlm.nih.gov/12223500/ Acesso em: 11 de jul. 2022.

ASTAH. **Astah Professional: UML, ER, DFD & Flowchart Software** - Astah. Disponível em: https://astah.net/products/astah-professional/. Acesso em: 11 de set. 2022.

BECK, K. et al. **The Agile Manifesto**. Agile Alliance. 2001. Disponível em: https://www.agilealliance.org/agile101/the-agile-manifesto/. Acesso em: 02 de jun. 2023.

BEEDLE, M. A. et al. **SCRUM: An extension pattern language for hyperproductive software development**. 2000. Disponível em: https://www.researchgate.net/publication/2464945_SCRUM_An_extension_pattern_language_for_hyperproductive_software_development. Acesso em: 18 set. 2022.

BERWICK, D. M.; LEAPE. L. L. **Reducing errors in medicine**. BMJ. 1999. Disponível em: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1116253/. Acesso em: 11 de jul. 2022.

BIBLUS. **WBS (Work Breakdown Structure), o que é e como se utiliza**. Disponível em: https://biblus.accasoftware.com/ptb/wbs-work-breakdown-structure-o-que-e-e-como-se-utiliza/. Acesso em: 11 de set. 2022.

BLOCH, K. V; MELO, A. N; NOGUEIRA, A. R. **Prevalência da adesão ao tratamento anti-hipertensivo em hipertensos resistentes e validação de três métodos indiretos de avaliação da adesão**. Cadernos de Saúde Pública, Rio de Janeiro, v. 24, n. 12, p.2979-2984, 2008. Disponível em: https://www.scielo.br/j/csp/a/bPNxQJ49ZZbzzqfSSDmPqLF/abstract/?lang=pt/. Acesso em: 10 de jun. 2023.

BYERS, C. **Sensores e Atuadores IoT**. 2018. Disponível em: https://embarcados.com.br/sensores-e-atuadores-iot/. Acesso: em 11 de set. 2022.

CLARK, M. **O que é o React Native Framework?** 12 dez. 2020. Disponível em: https://blog.back4app.com/pt/o-que-e-o-react-native-framework/. Acesso em: 10 de agosto de 2022.

CONSELHO NACIONAL DE SAÚDE. **Consumo de medicamentos: um autocuidado perigoso**. 2005. Disponível em: http://www.conselho.saude.gov.br/ultimas_noticias/2005/medicamentos.html. Acesso em: 02 de ago. 2022.

DE MELO, D. O; RIBEIRO, E.; STORPIRTIS, S. **A importância e a história dos estudos de utilização de medicamentos**. Revista Brasileira de Ciências Farmacêuticas Brazilian Journal of Pharmaceutical Sciences, v. 42, 2006. Disponível em: https://bvsms.saude.gov.br/bvs/is_digital/is_0207/pdfs/IS27(2)041.pdf. Acesso em: 27 de jul. 2022.

DE VRIES, E. M; RAMRATTAN M. A; SMORENBURG S. M; GOUMA D. J; BOERMEESTER M. A. **The incidence and nature or in-hospital adverse events: a systematic review. Qual Saf Health Care**. 2008.

DEVMEDIA. **Introdução ao Visual Studio Code**. 22 abr. 2016. Disponível em: https://www.devmedia.com.br/introducao-ao-visual-studio-code/34418. Acesso em: 10 de agosto de 2022.

DOS ANJOS, L. F. R. **EVOLUÇÃO DO JAVASCRIPT EM APLICAÇÕES MULTIPLATAFORMA**: Como projetos podem se beneficiar dos frameworks e bibliotecas disponíveis. Trabalho de Conclusão de Curso. Instituto Federal de Santa Catarina, Florianópolis, 2017. Disponível em: https://repositorio.ifsc.edu.br/bitstream/handle/123456789/1019/20190311_TCC_LuizFelipeDosAnjos.pdf. Acesso em: 11 de jul. 2022.

DOS REIS, F. **Introdução aos Sistemas Embarcados**. 2015. Disponível em: http://www.bosontreinamentos.com.br/eletronica/eletronica-geral/introducao-aos-sistemas-embarcados/. Acesso em: 12 de set. 2022.

DÜÜNA, Karl; RASHID, Fahmida Y. (Ed.). **Secure Your Node.js Web Application**. Raleigh: Pragmatic Programmers LLC, 2016.

ELETROGATE. 2022. **Sensores e Módulos**. Disponível em: https://www.eletrogate.com/sensores-e-modulos. Acesso em: 11 de set. 2022.

EMPRESÔMETRO. **Scrum: o método ágil mais utilizado no mundo corporativo. 2020**. Disponível em: https://blog.empresometro.com.br/scrum-o-metodo-agil-mais-utilizado-no-mundo-corporativo/. Acesso em: 17 de agosto de 2022.

ESPINHA, R. G. **Gráfico de Gantt: o que é, para que serve e como montar do zero**. 30 dez. 2021. Disponível em: https://artia.com/blog/grafico-de-gantt-o-que-e-para-que-serve-e-como-montar-o-seu/. Acesso em: 11 de set. 2022.

EWALLY. **Back-end: O Que É, Para Que Serve e Quais Suas Linguagens?** 06 set. 2021. Disponível em: https://www.ewally.com.br/blog/ajudando-sua-empresa/backend/. Acesso em: 11 de set. 2022.

EXPO. **FAQ - A list of common questions and limitations about Expo and related services**. Disponível em: https://docs.expo.dev/introduction/faq/. Acesso em: 10 de agosto de 2022.

FERNANDES, D. **Expo: o que é, para que serve e quando utilizar?** 27 mar. 2018. Disponível em: https://blog.rocketseat.com.br/expo-react-native/. Acesso em: 10 de agosto de 2022.

FLANAGAN, David. **JavaScript: O Guia Definitivo**. 6. ed. Porto Alegre: Bookman Editora Ltda, 2013. Tradução de: João Eduardo Nóbrega Tortello.

FOWLER, M. **UML Essencial: Um Breve Guia para a Linguagem-padrão de Modelagem de Objetos**. 3a Edição. Bookman, Porto Alegre, 2005.

GARCIA, F. D. **Introdução aos sistemas embarcados e microcontroladores**. 2019. Disponível em: https://embarcados.com.br/sistemas-embarcados-e-microcontroladores/. Acesso em: 16 de set. 2022.

GITHUB. Disponível em: https://github.com/. Acesso em: 11 de set. 2022.

HELENA, E. T S. **Adesão ao tratamento farmacológico de pacientes com hipertensão arterial em unidades de saúde da família em Blumenau, SC**. 2007. Tese (Doutorado em Medicina Preventiva) - Faculdade de Medicina, Universidade de São Paulo, São Paulo, 2007. Disponível em: https://www.teses.usp.br/teses/disponiveis/5/5137/tde-17022009-113221/pt-br.php. Acesso em: 10 de jun. 2023.

IQVIA INSTITUTE. **Global Medicine Spending and Usage Trends: Outlook to 2025**. 28 de abr. 2021. Disponível em: https://www.iqvia.com/insights/the-iqvia-institute/reports/global-medicine-spending-and-usage-trends-outlook-to-2025. Acesso em: 10 de jun. 2023.

KOHN, L. T. et al. **To Error is human: building a safer health system**. Washington: Committee on Quality of Health Care in America, National Academy of Institute of Medicine, 2001. Disponível em: https://www.ncbi.nlm.nih.gov/books/NBK225182/. Acesso em: 11 de jul. 2022.

KRIGER, B. **O que é front end, para que serve e como aprender front end**. 2020. Disponível em: https://kenzie.com.br/blog/front-end/. Acesso em: 10 de set. 2022.

LAUGENI, Fernando Piero; MARTINS, Petrônio Garcia. **Administração da Produção**. São Paulo: Saraiva, 2006.

LISBOA, V. G. C. **Protoboard**, 2015. Disponível em: http://www.uel.br/pessoal/ernesto/arduino/00_Protoboard.pdf. Acesso em 11 de set. 2022.

LIVEFINE. **Automatic Pill Dispenser with Bluetooth**. Disponível em: https://www.livefineproducts.com/collections/smart-pill-dispensers/products/automatic-pill-dispenser-with-bluetooth%C2%AE. Acesso em: 11 de jul. 2022.

LOPES, J. DE O. PHP ou **TypeScript: uma comparação de duas linguagens para web pelas suas características**. Trabalho de Conclusão de Curso. Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul, 2016. Disponível em: http://atom.poa.ifrs.edu.br/index.php/php-ou-typescript-uma-comparacao-de-duas-linguagens-para-web-pelas-suas-caracteristicas. Acesso em: 19 de jul. 2022.

LYNCH, S. S. **Manual MSD: Adesão ao esquema terapêutico**. Jul. 2022. Disponível em: https://www.msdmanuals.com/pt-br/profissional/farmacologia-cl%C3%ADnica/fatores-que-afetam-a-resposta-a-f%C3%A1rmacos/ades%C3%A3o-ao-esquema-terap%C3%AAutico. Acesso em: 11 de jun. 2023.

MCROBERTS, M. **Arduino Básico**. Novatec Editora, 2018. Disponível em: https://edisciplinas.usp.br/pluginfile.php/4287597/mod_resource/content/2/Ardu%C3%ADno%20B%C3%A1sico%20-%20Michael%20McRoberts.pdf. Acesso em: 19 jul. 2022.

MELO, D. O; RIBEIRO, E; STORPIRTIS, S. A importância e a história dos estudos de utilização de medicamentos. Revista Brasileira de Ciências Farmacêuticas, 42 (4), 2006. Disponível em: https://www.revistas.usp.br/rbcf/issue/view/3613. Acesso em: 11 de jun. 2023.

MINHAJ, A. **MeDuino: Smart Automatic Medicine Reminder**. Disponível em: https://create.arduino.cc/projecthub/ashraf_minhaj/meduino-smart-automatic-medicine-reminder-84a4a8?ref=part&ref_id=8233&offset=63. Acesso em: 11 de jul. 2022.

MINISTÉRIO DA SAÚDE. **Síntese de evidências para políticas de saúde: Adesão ao tratamento medicamentoso por pacientes portadores de doenças crônicas**. Ministério da Saúde, Brasília, 2016. Disponível em: https://pesquisa.bvsalud.org/portal/resource/pt/biblio-971867. Acesso em: 11 de jun. 2023.

NANCE, Christofer. **TypeScript Essentials: Develop large scale responsive web applications with TypeScrit**. Birminghan - UK: Packt Publishing, 2014. ISBN 978-178398-576-0. Acesso em 11 de jul. 2022

PAYNE, R; FRANKLIN, B, D; SLIGHT, S; AVERY, A. **Medication Errors: Technical Series on Safer Primary Care**. 2006.

PEREIRA, C. R. **Node.js: Aplicações web real-time com Node.js**. São Paulo: Casa do Código, 2013. Acesso em: 19 de jul. 2022.

POZZEBOM, R. **O que é GitHub?** 07 jul. 2015. Disponível em: https://www.oficinadanet.com.br/post/14791-o-que-github. Acesso em: 10 de agosto de 2022.

POZZEBOM, R. **O que são sistemas embarcados?** 2014. Disponível em: https://www.oficinadanet.com.br/post/13538-o-que-sao-sistemas-embarcados. Acesso: em 11 de set. 2022.

PRESSMAN, Roger S. **Engenharia de Software: Uma abordagem profissional**. Bookman, Porto Alegre, 2011.

RÉGIS, L. **Remédios têm sido tratados como bens de consumo', diz doutor em toxicologia da USP**. Estadão, São Paulo. 10 dez. 2016. Disponível em:

https://infograficos.estadao.com.br/focas/tanto-remedio-para-que/checkup-5.php. Acesso em: 10 de jun. 2023.

REMONDI, F. A.; CABRERA, M. A. S.; SOUZA, R. K. T. DE. **Não adesão ao tratamento medicamentoso contínuo: prevalência e determinantes em adultos de 40 anos e mais**. Cadernos de Saúde Pública, v. 30, n. 1, p. 126–136, 2014. Disponível em: https://www.scielo.br/j/csp/a/89w4bykTcKBwFLZRcSJ8cTF/. Acesso em: 12 de jun. 2023.

RODRIGUES, G. V. **Tutorial caixa time**. 05 jul. 2017. Disponível em: https://autocorerobotica.blog.br/tutorial-caixa-time/. Acesso em: 11 de jul. 2022.

SBROCCO, J.H. T. C.; MACEDO, P. C. de. **Metodologias Ágeis: Engenharia de Software sob medida**, 2012.

SINDUSFARMA. **Perfil da indústria farmacêutica e aspectos relevantes do setor**. Mai. 2022. Disponível em: https://sindusfarma.org.br/uploads/files/229d-gerson-almeida/Publicacoes_PPTs/PERFIL_IND_FARMACEUTICA_22_PORT.pdf. Acesso em: 02 de ago. 2022.

SOLOMON K. **What is Trello used for? Our fave project management software explained**. 15 abr. 2022. Disponível em: https://blog.trello.com/what-is-trello-used-for. Acesso em: 09 de agosto de 2022.

SOUZA, M. L. P.; GARNELO, L. **"É muito dificultoso!": etnografia dos cuidados a pacientes com hipertensão e/ou diabetes na atenção básica, em Manaus, Amazonas, Brasil**. Cadernos de Saúde Pública, Rio de Janeiro, v. 1, p. 91-99, 2008. Disponível em: https://www.scielo.br/j/csp/a/WtZfjgNTnxrSXCCNrP3cr5J/. Acesso em: 10 de jun. 2023.

TELES, A. S. et al. **Papel dos medicamentos nas intoxicações causadas por agentes químicos em municípios da Bahia, no período de 2007 a 2010**. 2013. Acesso em: 19 de jul. 2022.

THOMSEN, A. **O que é Arduino, para que serve e primeiros passos**. 2014. Disponível em: https://www.filipeflop.com/blog/o-que-e-arduino/. Acesso em: 10 de agosto de 2022.

TOTVS. **Front end: O que é, como funciona e qual a importância**. 14 jul. 2021. Disponível em: https://www.totvs.com/blog/developers/front-end/. Acesso em: 10 de set. 2022.

TRAN, N. et al. **Patient reminder systems and asthma medication adherence: a systematic review**. Journal of Asthma, London, v. 51, n. 5, p. 536-543, 2014.

TRELLO. **About Trello: What's behind the boards**. Disponível em: https://trello.com/about. Acesso em: 09 de agosto de 2022.

VISUAL STUDIO CODE. **Visual Studio Code: Editor de código fonte**. Disponível em: https://code.visualstudio.com/. Acesso em: 11 de set. 2022.

WORLD HEALTH ORGANIZATION. **Medication errors**. 2016. World Health Organization. Disponível em: https://apps.who.int/iris/handle/10665/252274. Acesso em: 11 de jun. 2023.

WU, W. **React Native vs Flutter, cross-platform mobile application frameworks**. 01 March 2018 About Arduino. Disponível em: https://www.theseus.fi/handle/10024/146232. Acesso em: 19 jul. 2022.

APÊNDICE A – LISTA DE REQUISITOS

1.1 REQUISITOS DO ARDUINO

RF001 – Enviar alertas visuais e sonoros

O Arduino deve notificar o usuário de forma visual e sonora nos horários agendados para cada medicamento cadastrado no aplicativo. Para que o usuário receba esses alertas, é necessário que o remédio cadastrado esteja vinculado a uma gaveta no aplicativo.

RF002 – Registrar de abertura/fechamento de gaveta

O Arduino deve registrar informações de horário de abertura/fechamento de cada gaveta, e posteriormente enviar para esses registros para o aplicativo.

RF003 – Registrar não-abertura de gavetas no horário agendado

Caso não haja registro de abertura da gaveta no horário agendado, o Arduino deve enviar ao aplicativo a informação de que a gaveta não foi aberta.

1.2 REQUISITOS DO APLICATIVO

RF001 – Cadastrar medicamento

O Usuário pode cadastrar um novo medicamento, informando nome, horário e data inicial, quantidade de remédios e de dias, dosagem e intervalo.

RF002 – Editar medicamento

O Usuário pode editar um medicamento já cadastrado, alterando qualquer informação desejada.

RF003 – Remover medicamento

O Usuário pode excluir um medicamento cadastrado. Ao excluir um medicamento que esteja em uma gaveta, ele também é desvinculado da gaveta.

RF004 – Visualizar gavetas

O Usuário pode acessar as gavetas disponíveis pelo menu. Ele pode escolher uma gaveta para vincular um medicamento ou verificar informações de uma gaveta já ocupada.

RF005 – Vincular medicamento

O Usuário vincula um remédio previamente cadastrado a uma gaveta disponível, alterando o estado da gaveta para ocupada.

RF006 – Desvincular medicamento

O Usuário desvincula um medicamento de uma gaveta, caso ela esteja ocupada. Assim a gaveta torna-se livre novamente para receber um novo medicamento.

RF007 – Visualizar relatório de histórico

O Usuário pode acessar a tela de histórico e visualizar os últimos medicamentos tomados. Para cada registro, o aplicativo fornece as seguintes informações: nome do remédio, horário agendado e o horário de abertura da gaveta. Caso não exista registro de abertura para determinado horário agendado, o aplicativo irá informar que o medicamento não foi ingerido.

RF008 – Filtrar histórico

O usuário pode filtrar o histórico, podendo escolher um medicamento e/ou um período específico.

RF009 – Visualizar Dashboard

O Usuário tem acesso a um Dashboard contendo informações de gerenciamento dos medicamentos ao acessar a página inicial do aplicativo. Os dados disponíveis são: horário do próximo medicamento, sequência de dias de medicamentos tomados no horário correto, quantidade de medicamentos não tomados, nome do último medicamento ingerido e um gráfico contendo informações quanto a quantidade de medicamentos (compridos, capsulas, etc.) em cada gaveta.

RF010 – Visualizar calendário

O Usuário pode acessar um calendário a partir do menu. Ao escolher uma data específica, ele poderá visualizar os horários e os medicamentos que estão agendados para dia selecionado. O aplicativo só irá mostrar remédios que estejam vinculados a uma gaveta.

RF011 – Personalizar configurações

O Usuário acessa as configurações do aplicativo a partir do menu, podendo: cadastrar um contato e enviar relatórios para o contato cadastrado.

RF012 – Cadastrar contato

O Usuário, ao acessar o menu de configurações, poderá cadastrar o número de celular de um contato, que poderá receber notificações via mensagem do andamento do tratamento do paciente.

RF013 – Enviar relatórios

O Usuário, a partir do menu de configurações, poderá enviar ao número do contato cadastrado informações sobre o andamento de seu tratamento via Whatsapp.

RNF014 – Enviar de calendário de medicamentos para o Arduino

Após o Usuário vincular um medicamento a uma gaveta, o aplicativo deverá enviar as informações de agendamento para o Arduino.

RNF015 – Notificar Arduino alterações de medicamentos

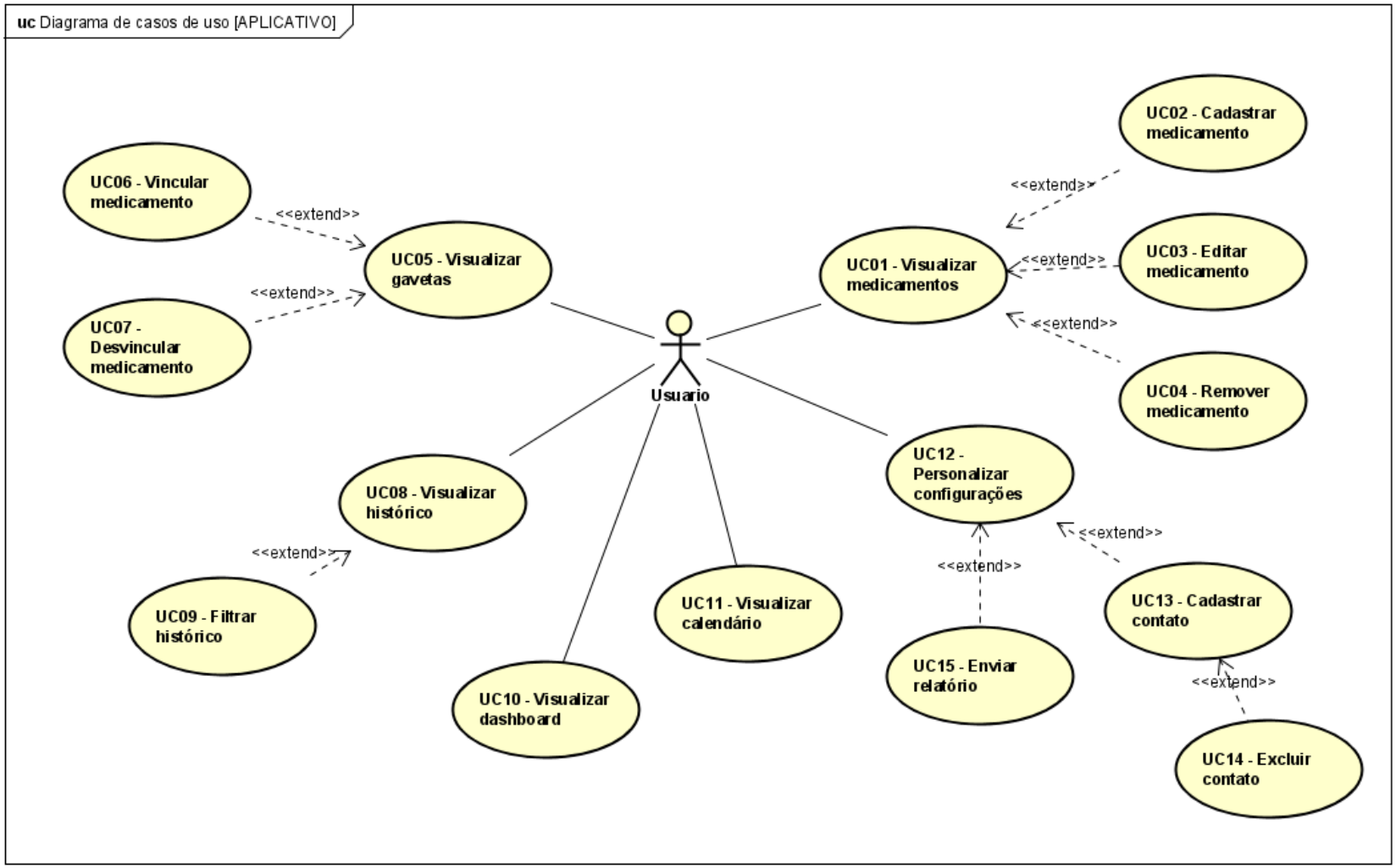
Caso o usuário desvincule ou altere informações de um remédio da gaveta, o Arduino deve ser notificado pelo aplicativo.

RNF016 – Notificar usuário de atraso nos medicamentos

Caso o aplicativo seja notificado pelo Arduino de que a gaveta não foi aberta no horário correspondente, o aplicativo deve enviar ao usuário uma notificação de atraso no medicamento.

APÊNDICE B – DIAGRAMA DE CASOS DE USO DO APLICATIVO

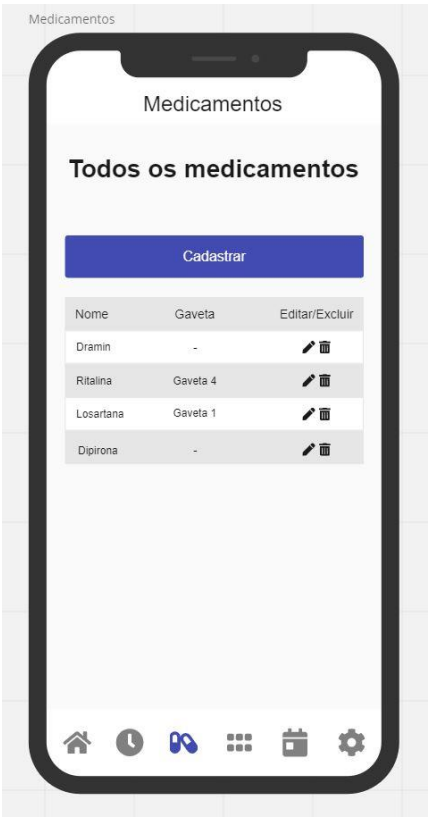
FIGURA 37 - DIAGRAMA DE CASOS DE USO DO APLICATIVO



FONTE: OS AUTORES (2023).

APÊNDICE C - ESPECIFICAÇÃO DE CASOS DE USO DO APLICATIVO

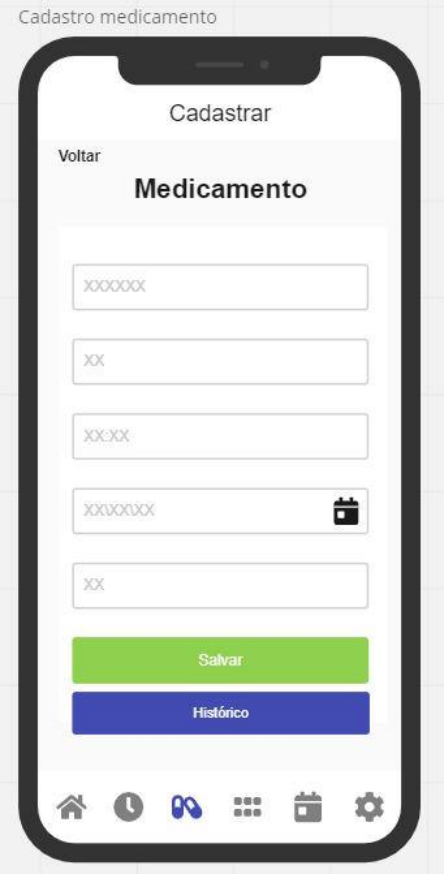
TABELA 4 - UC01 - VISUALIZAR MEDICAMENTOS

Nome do UC	**UC01 – Visualizar medicamentos**
Descrição	Nesse caso de uso o usuário visualiza a lista de medicamentos cadastrados no aplicativo.
Data View	
Pré condições	-
Pós condições	-
Fluxo principal	1. Usuário clica no botão “medicamentos”, 2. Sistema renderiza lista de medicamentos cadastrados.
Fluxo alternativo	A1: Adicionar medicamento 1. Usuário clica no botão “cadastrar”. 2. Sistema executa caso de uso Cadastrar Medicamento. A2: Editar medicamento 1. Usuário clica no botão “Editar”. 2. Sistema executa caso de uso Editar Medicamento.

	A3: Remover medicamento 1. Usuário clica no botão “Remover”. 2. Sistema executa caso de uso Remover Medicamento.
Fluxo de exceção	-
Regra de negócio	-

FONTE: OS AUTORES (2023).

TABELA 5 - UC02 - CADASTRAR MEDICAMENTO

Nome do UC	**UC02 – Cadastrar medicamento**
Descrição	Nesse caso de uso o usuário realiza o cadastro de um novo medicamento.
Data View	
Pré condições	-
Pós condições	-
Fluxo principal	1. Sistema renderiza formulário de cadastro. 2. Usuário preenche os campos com informações do medicamento e clica em “Salvar”. 3. Sistema verifica regra R1. 4. Sistema grava o novo medicamento.
Fluxo alternativo	A1: Usuário deseja selecionar um medicamento previamente cadastrado. 1. Usuário clica no botão de histórico. 2. Sistema lista todos os medicamentos já cadastrados (ativos e inativos). 3. Usuário seleciona medicamento.

	4. Sistema preenche formulário com os dados do medicamento.
Fluxo de exceção	E1: Campos não preenchidos 1. Sistema mostra mensagem de erro “Preencha todos os campos”.
Regra de negócio	R1: Todos os campos devem estar preenchidos.

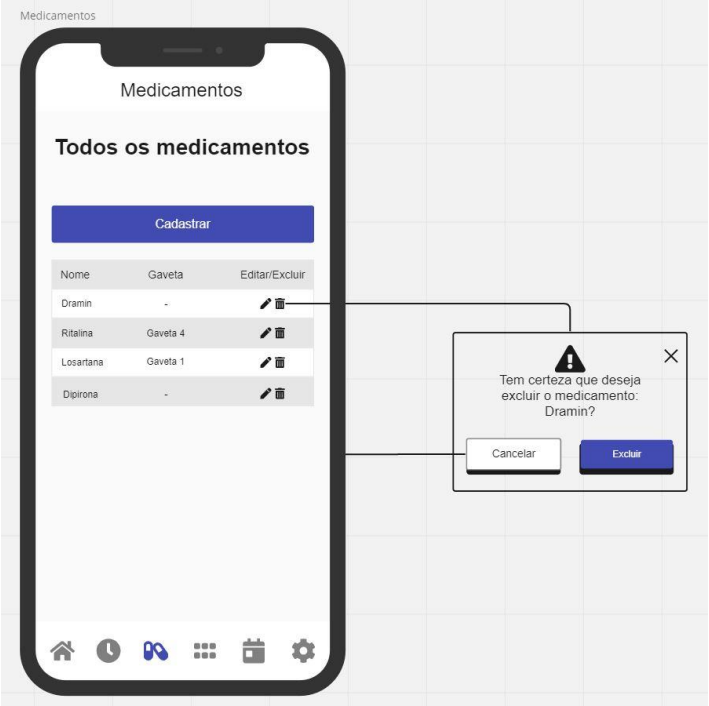
FONTE: OS AUTORES (2023).

TABELA 6 - UC03 - EDITAR MEDICAMENTO

Nome do UC	**UC03 – Editar medicamento**
Descrição	Nesse caso de uso o usuário modifica informações de um medicamento já cadastrado no sistema.
Data View	
Pré condições	-
Pós condições	-
Fluxo principal	1. Sistema renderiza formulário preenchido com as informações do medicamento. 2. Usuário modifica as informações desejadas e clica em “Salvar”. 3. Sistema verifica regra R1. 4. Sistema grava as alterações.
Fluxo alternativo	A1: Medicamento está vinculado a uma gaveta 1. Sistema envia as novas informações do calendário do medicamento para o Arduino.
Fluxo de exceção	E1: Campos não preenchidos 1. Sistema mostra mensagem de erro “Preencha todos os campos”.
Regra de negócio	R1: Todos os campos devem estar preenchidos.

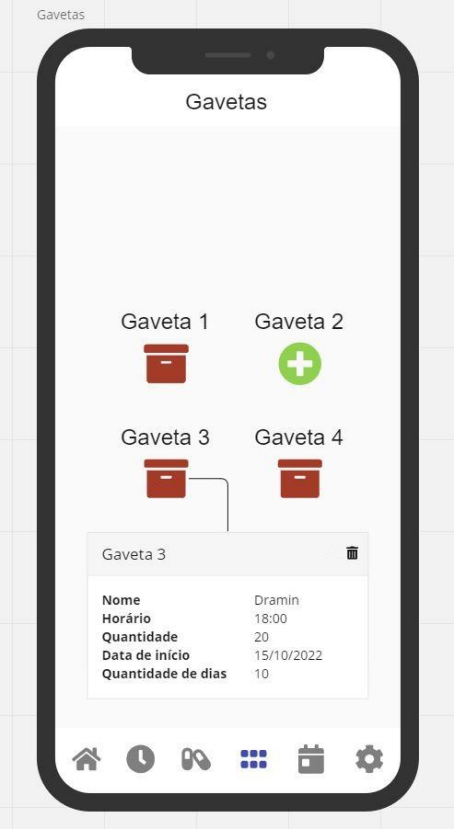
FONTE: OS AUTORES (2023).

TABELA 7 - UC04 - REMOVER MEDICAMENTO

Nome do UC	**UC04 – Remover medicamento**
Descrição	Nesse caso de uso o usuário deleta um medicamento cadastrado.
Data View	
Pré condições	-
Pós condições	-
Fluxo principal	1. Usuário clica no ícone “Deletar”. 2. Sistema exibe caixa de confirmação de remoção. 3. Usuário clica em “Confirmar”. 4. Sistema verifica existência de vínculo com uma gaveta. 5. Sistema atualiza o status do medicamento como inativo.
Fluxo alternativo	A1: Existência de vínculo com gaveta 1. Sistema remove o vínculo do medicamento com a gaveta. 2. Sistema envia informação para o Arduino remover o remédio da gaveta.
Fluxo de exceção	-
Regra de negócio	-

FONTE: OS AUTORES (2023).

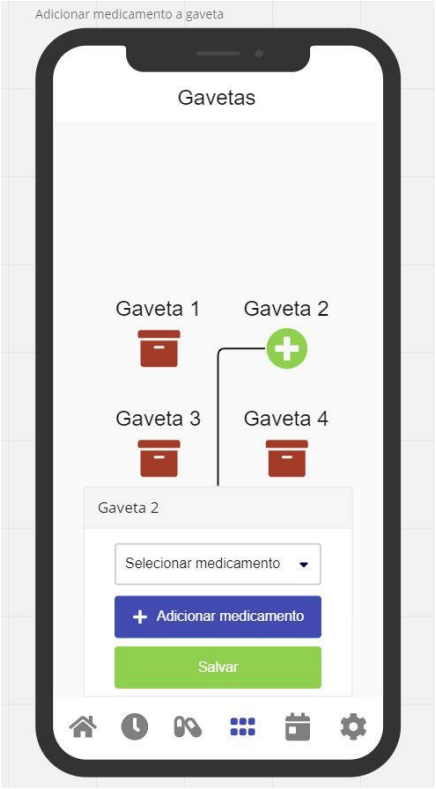
TABELA 8 - UC05 - VISUALIZAR GAVETAS

Nome do UC	**UC05 – Visualizar gavetas**
Descrição	Nesse caso de uso o usuário visualiza a página de gavetas.
Data View	
Pré condições	-
Pós condições	-
Fluxo principal	1. Usuário clica no botão “Gavetas”. 2. Sistema renderiza a tela de gavetas.
Fluxo alternativo	A1: Vincular remédio a gaveta 1. Usuário seleciona uma gaveta vazia. 2. Sistema executa caso de uso Vincular Medicamento. A2: Desvincular remédio da gaveta. 1. Usuário seleciona uma gaveta ocupada. 2. Sistema executa caso de uso Desvincular Medicamento. A3: Visualizar informações do medicamento de uma gaveta ocupada. 1. Usuário seleciona uma gaveta ocupada.

	2. Sistema mostra dados do medicamento vinculado a gaveta escolhida.
Fluxo de exceção	-
Regra de negócio	-

FONTE: OS AUTORES (2023).

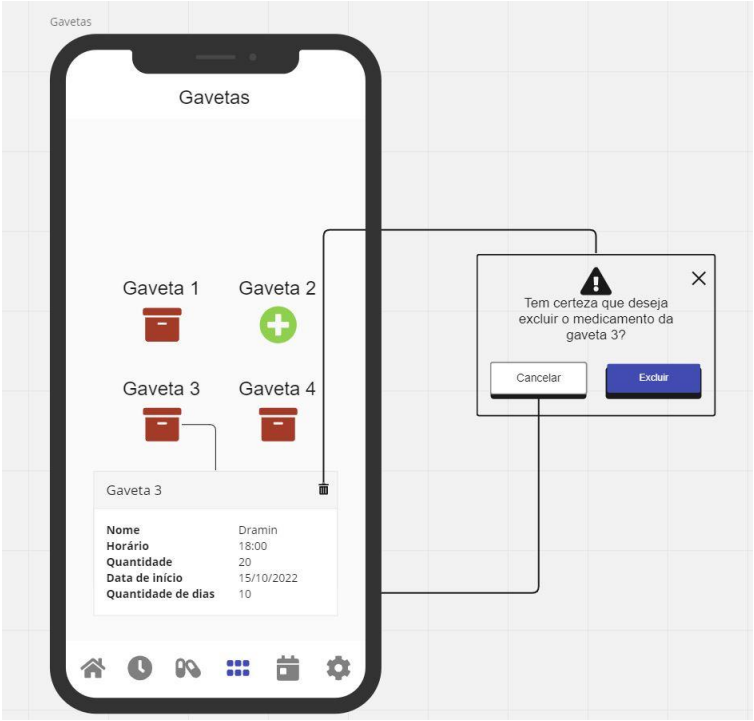
TABELA 9 - UC06 - VINCULAR MEDICAMENTO

Nome do UC	**UC06 – Vincular medicamento**
Descrição	Nesse caso de uso o usuário vincula um medicamento cadastrado a uma gaveta vazia.
Data View	
Pré condições	O remédio a ser vinculado deve ter sido previamente cadastrado na página de medicamentos. A gaveta selecionada deve estar vazia.
Pós condições	-
Fluxo principal	1. Usuário seleciona uma gaveta vazia. 2. Sistema renderiza a lista de medicamentos aplicando a regra R1. 3. Usuário seleciona o medicamento desejado e clica em “Salvar”. 4. Sistema salva o vínculo da gaveta com o medicamento. 5. Sistema envia calendário do medicamento selecionado para o Arduino.
Fluxo alternativo	A1: Medicamento não cadastrado 1. Usuário clica no botão “Adicionar medicamento”

	2. Sistema redireciona para a página de medicamentos e executa caso de uso Adicionar Medicamento.
Fluxo de exceção	-
Regra de negócio	R1: Apenas medicamentos que não estão em uma gaveta são listados.

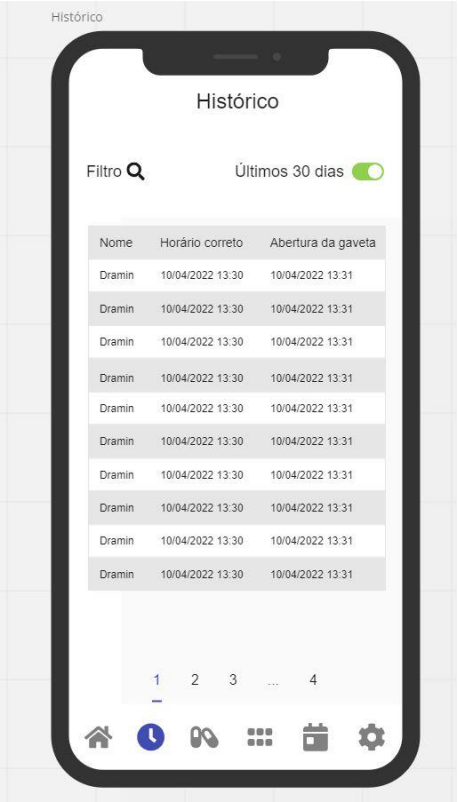
FONTE: OS AUTORES (2023).

TABELA 10 - UC07 - DESVINCULAR MEDICAMENTO

Nome do UC	**UC07 – Desvincular medicamento**
Descrição	Nesse caso de uso o usuário desvincula um medicamento de uma gaveta.
Data View	
Pré condições	-
Pós condições	-
Fluxo principal	1. Usuário seleciona uma gaveta ocupada. 2. Sistema mostra dados do medicamento vinculado a gaveta escolhida. 3. Usuário clica no ícone “Excluir”. 4. Sistema exibe caixa de confirmação de remoção. 5. Usuário clica em “Excluir”. 6. Sistema remove o vínculo do remédio com a gaveta. 7. Sistema envia informação para o Arduino remover o remédio da gaveta.
Fluxo alternativo	-
Fluxo de exceção	-
Regra de negócio	-

FONTE: OS AUTORES (2023).

TABELA 11 - UC08 - VISUALIZAR HISTÓRICO

Nome do UC	**UC08 – Visualizar histórico**
Descrição	Nesse caso de uso o usuário visualiza a tela de históricos de medicamentos consumidos.
Data View	
Pré condições	-
Pós condições	-
Fluxo principal	1. Usuário clica no ícone de “Histórico”. 2. Sistema busca informações de abertura de gavetas do Arduino e atualiza a base de dados de histórico. 3. Sistema renderiza tela com dados do histórico dos últimos 30 dias ordenado por data.
Fluxo alternativo	A1: Filtrar histórico 1. Usuário clica em “Filtrar”. 2. Sistema executa caso de uso Filtrar Histórico.
Fluxo de exceção	-
Regra de negócio	-

FONTE: OS AUTORES (2023).

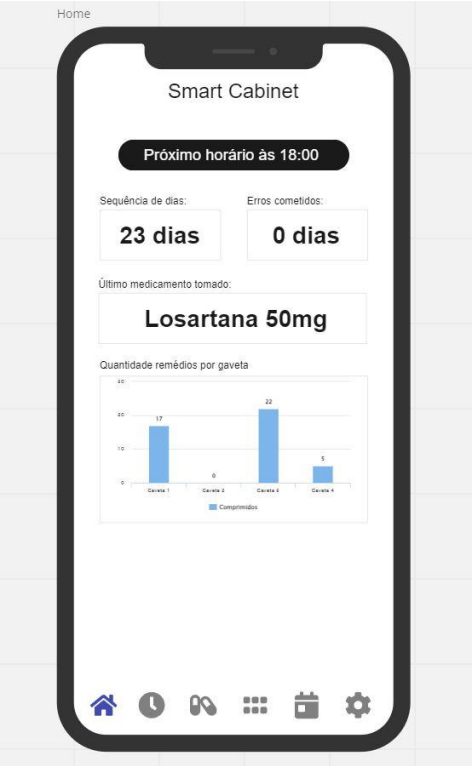
TABELA 12 - UC09 - FILTRAR HISTÓRICO

Nome do UC	**UC09 – Filtrar histórico**
Descrição	Nesse caso de uso o usuário aplica os filtros desejados no histórico.
Data View	
Pré condições	-
Pós condições	-
Fluxo principal	1. Usuário clica em “Filtro” 2. Sistema renderiza tela com formulário de filtro. 3. Usuário preenche os campos de filtro. 4. Sistema verifica as regras R1 e R2. 5. Sistema renderiza o histórico com o filtro aplicado.
Fluxo alternativo	A1: Remover filtro 1. Usuário clica em “Remover filtro”. 2. Sistema remove o filtro e executa caso de uso Visualizar Histórico.
Fluxo de exceção	E1: Filtros inválidos 1. Sistema mostra mensagem de erro “Filtros inválidos”.
Regra de negócio	R1: Todos os campos devem ser preenchidos.

	R2: Para filtrar por período, data de início deve ser menor que a data fim.

FONTE: OS AUTORES (2023).

TABELA 13 - UC10 - VISUALIZAR DASHBOARD

Nome do UC	**UC10 – Visualizar Dashboard**
Descrição	Nesse caso de uso o usuário visualiza o dashboard na tela home do aplicativo.
Data View	
Pré condições	-
Pós condições	-
Fluxo principal	1. Usuário abre o aplicativo. 2. Sistema renderiza o dashboard com informações do tratamento do paciente.
Fluxo alternativo	-
Fluxo de exceção	-
Regra de negócio	-

FONTE: OS AUTORES (2023).

TABELA 14 - UC11 - VISUALIZAR CALENDÁRIO

Nome do UC	**UC11 – Visualizar calendário**
Descrição	Nesse caso de uso o usuário visualiza tela de calendário.
Data View	
Pré condições	-
Pós condições	-
Fluxo principal	1. Usuário clica no ícone “Calendário”. 2. Sistema renderiza o calendário. 3. Usuário seleciona a data desejada. 4. Sistema mostra os medicamentos e seus respectivos horários para o dia selecionado.
Fluxo alternativo	-
Fluxo de exceção	-
Regra de negócio	-

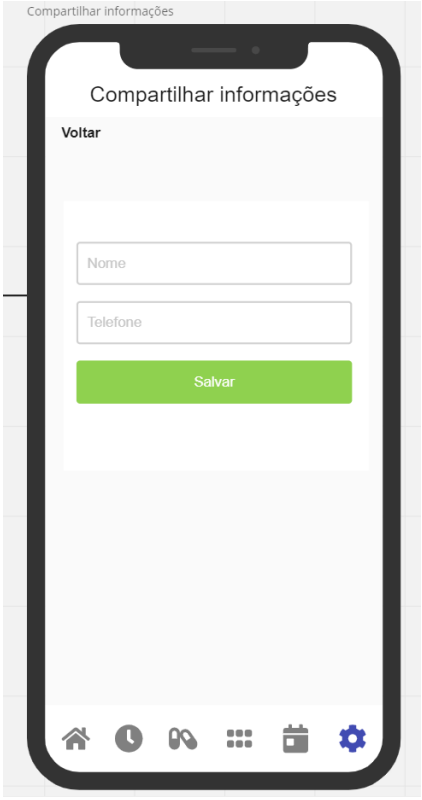
FONTE: OS AUTORES (2023).

TABELA 15 - UC12 - PERSONALIZAR CONFIGURAÇÕES

Nome do UC	**UC12 – Personalizar configurações**
Descrição	Nesse caso de uso o usuário acessa a tela de configurações do aplicativo.
Data View	
Pré condições	-
Pós condições	-
Fluxo principal	1. Usuário clica no ícone “Configurações”. 2. Sistema renderiza tela de configurações.
Fluxo alternativo	A1: Cadastrar contato 1. Usuário clica em “Contatos”. 2. Sistema executa o caso de uso Cadastrar Contato. A2: Enviar relatório ao contato 1. Usuário clica em “Enviar relatório”. 2. Sistema executa o caso de uso Enviar Relatório.
Fluxo de exceção	-
Regra de negócio	-

FONTE: OS AUTORES (2023).

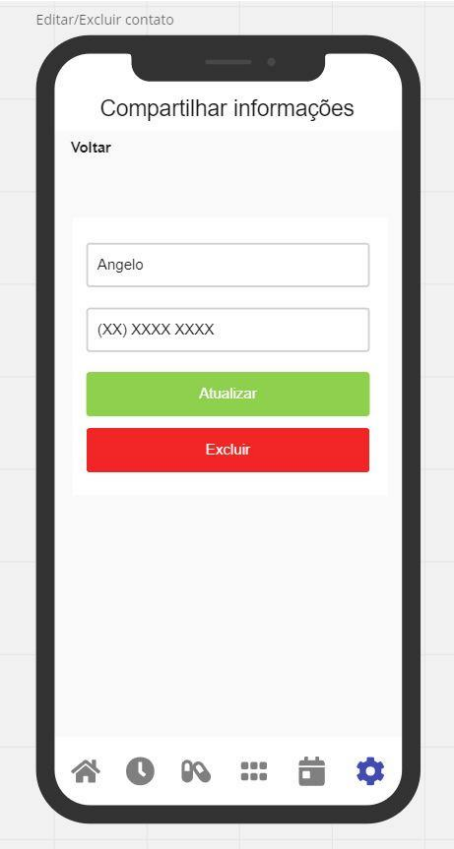
TABELA 16 - UC13 - CADASTRAR CONTATO

Nome do UC	**UC13 – Cadastrar contato**
Descrição	Nesse caso de uso o usuário cadastra o número de telefone de um contato.
Data View	
Pré condições	-
Pós condições	-
Fluxo principal	1. Usuário clica no botão “Contato”. 2. Sistema renderiza tela de cadastro de contato. 3. Usuário entra com as informações do contato e clica em “Salvar”. 4. Sistema verifica a regra R1. 5. Sistema salva o contato.
Fluxo alternativo	A1: Já possui um contato cadastrado 1. Sistema abre formulário de contato já preenchido. 2. Usuário faz as alterações desejadas no formulário e clica em “Salvar”. 3. Sistema verifica a regra R1. 4. Sistema atualiza o contato. A2: Excluir contato já cadastrado 1. Usuário clica em “Excluir”.

	2. Sistema executa o caso de uso Excluir Contato.
Fluxo de exceção	-
Regra de negócio	R1: Todos os campos devem ser preenchidos.

FONTE: OS AUTORES (2023).

TABELA 17 - UC14 - EXCLUIR CONTATO

Nome do UC	**UC14 – Excluir contato**
Descrição	Nesse caso de uso o usuário exclui o contato cadastrado.
Data View	
Pré condições	Deve haver um contato cadastrado.
Pós condições	-
Fluxo principal	1. Usuário clica no botão “Excluir”. 2. Sistema exibe caixa de confirmação de remoção. 3. Usuário clica em “Excluir”. 4. Sistema remove o contato.
Fluxo alternativo	-
Fluxo de exceção	-
Regra de negócio	-

FONTE: OS AUTORES (2023).

TABELA 18 - UC15 - ENVIAR RELATÓRIO

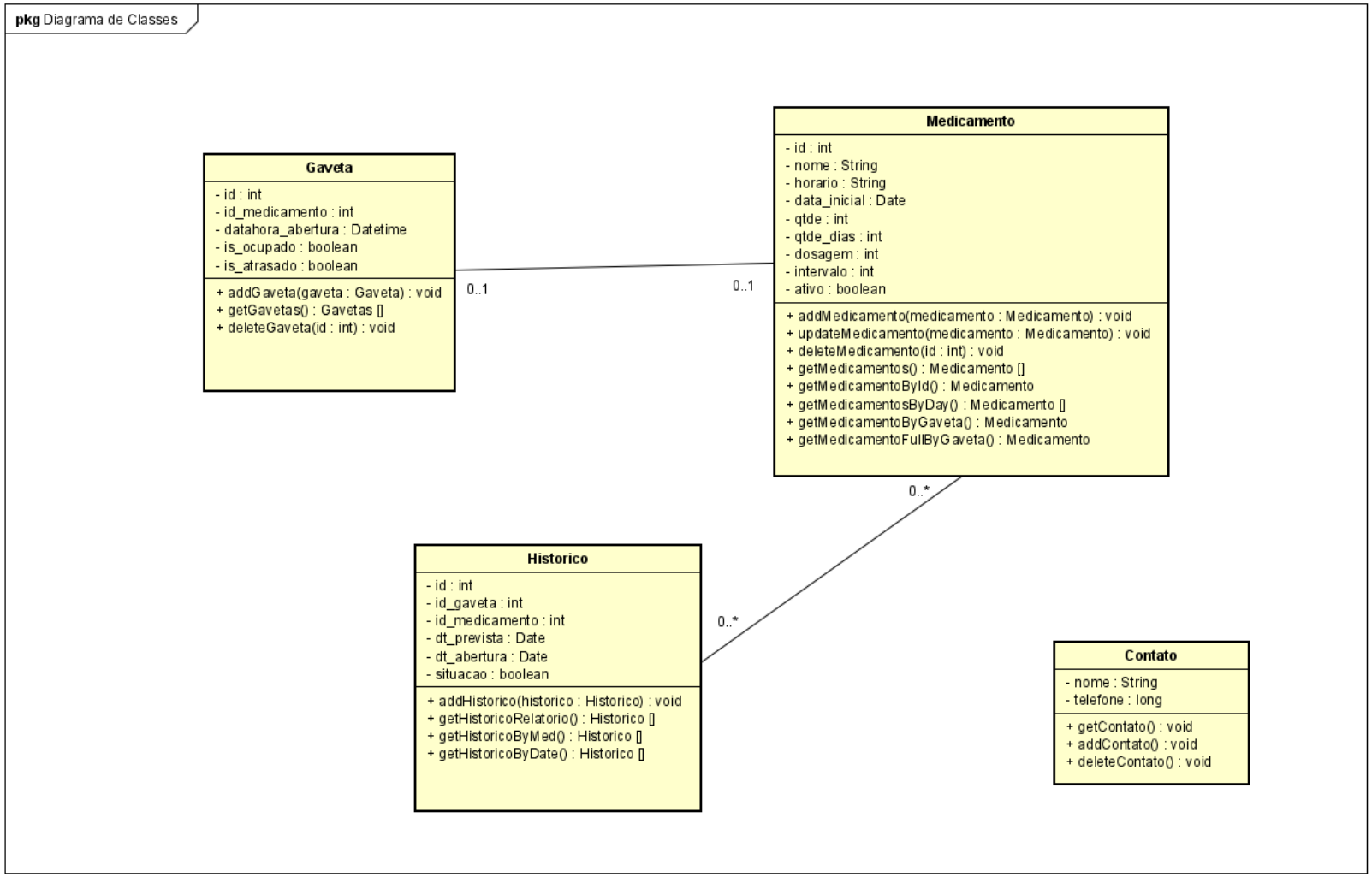
Nome do UC	**UC15 – Enviar relatório**
Descrição	Nesse caso de uso o usuário envia um relatório de um medicamento ao contato cadastrado.
Data View	
Pré condições	Deve haver um contato cadastrado. Deve haver pelo menos um medicamento em uma gaveta.
Pós condições	-
Fluxo principal	1. Usuário clica no botão “Enviar relatório”. 2. Sistema verifica exceção E1 e E2. 3. Sistema envia o relatório para o número cadastrado.
Fluxo alternativo	
Fluxo de exceção	E1: Não há medicamentos em gavetas 1. Sistema mostra alerta “não há medicamentos nas gavetas”. E2: Não há um contato cadastrado 1. Sistema mostra alerta “não há um contato cadastrado”.

Regra de negócio	-

FONTE: OS AUTORES (2023).

APÊNDICE D – DIAGRAMA DE CLASSES

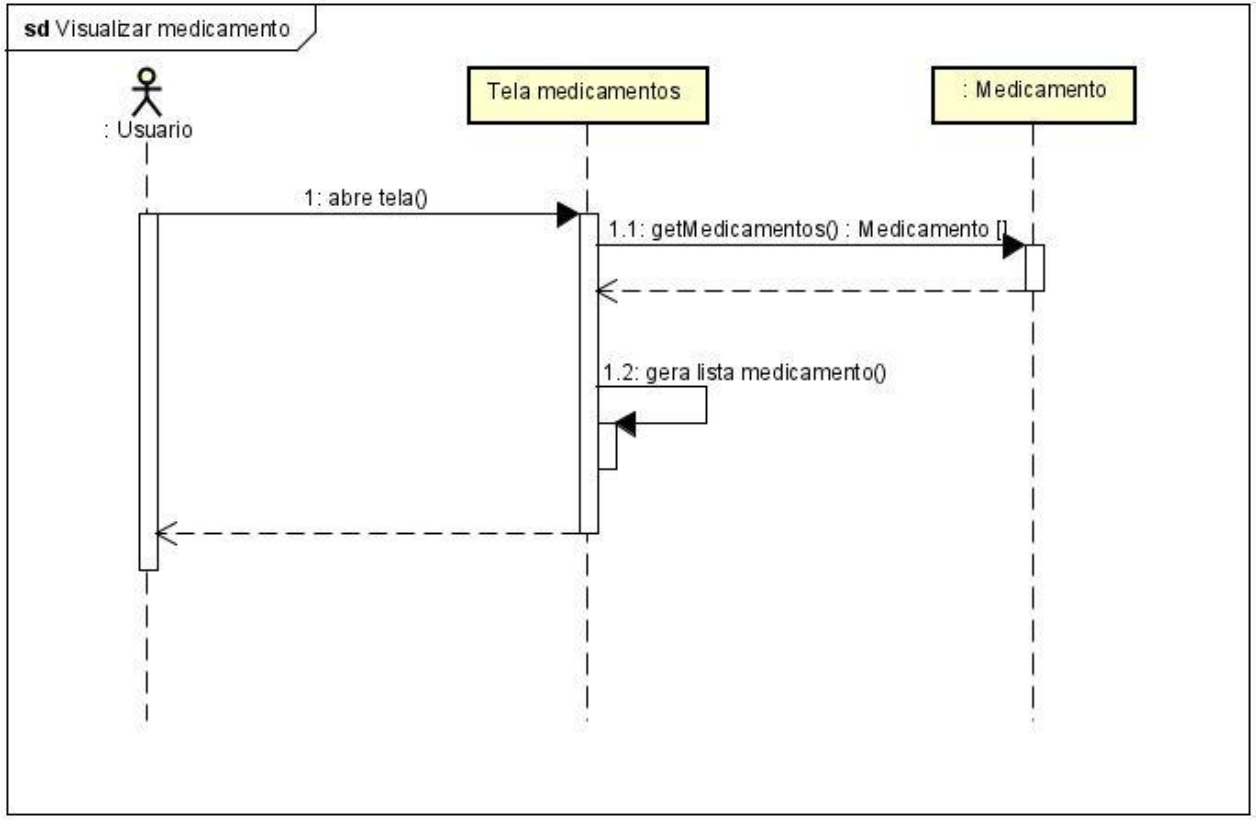
FIGURA 38 - DIAGRAMA DE CLASSES DO APLICATIVO



FONTE: OS AUTORES (2023).

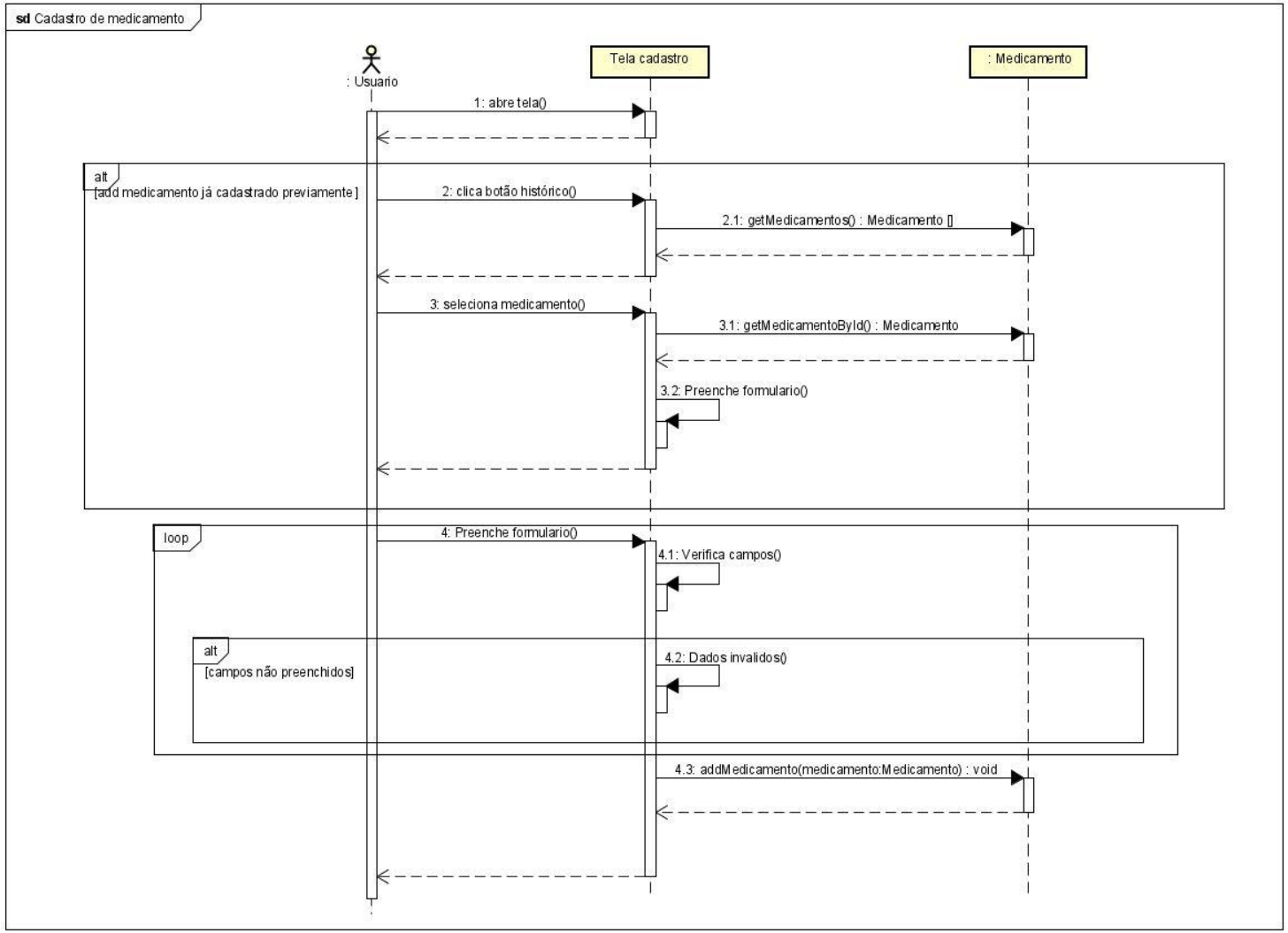
APÊNDICE E – DIAGRAMA DE SEQUÊNCIA

FIGURA 39 - DIAGRAMA DE SEQUÊNCIA VISUALIZAR MEDICAMENTO



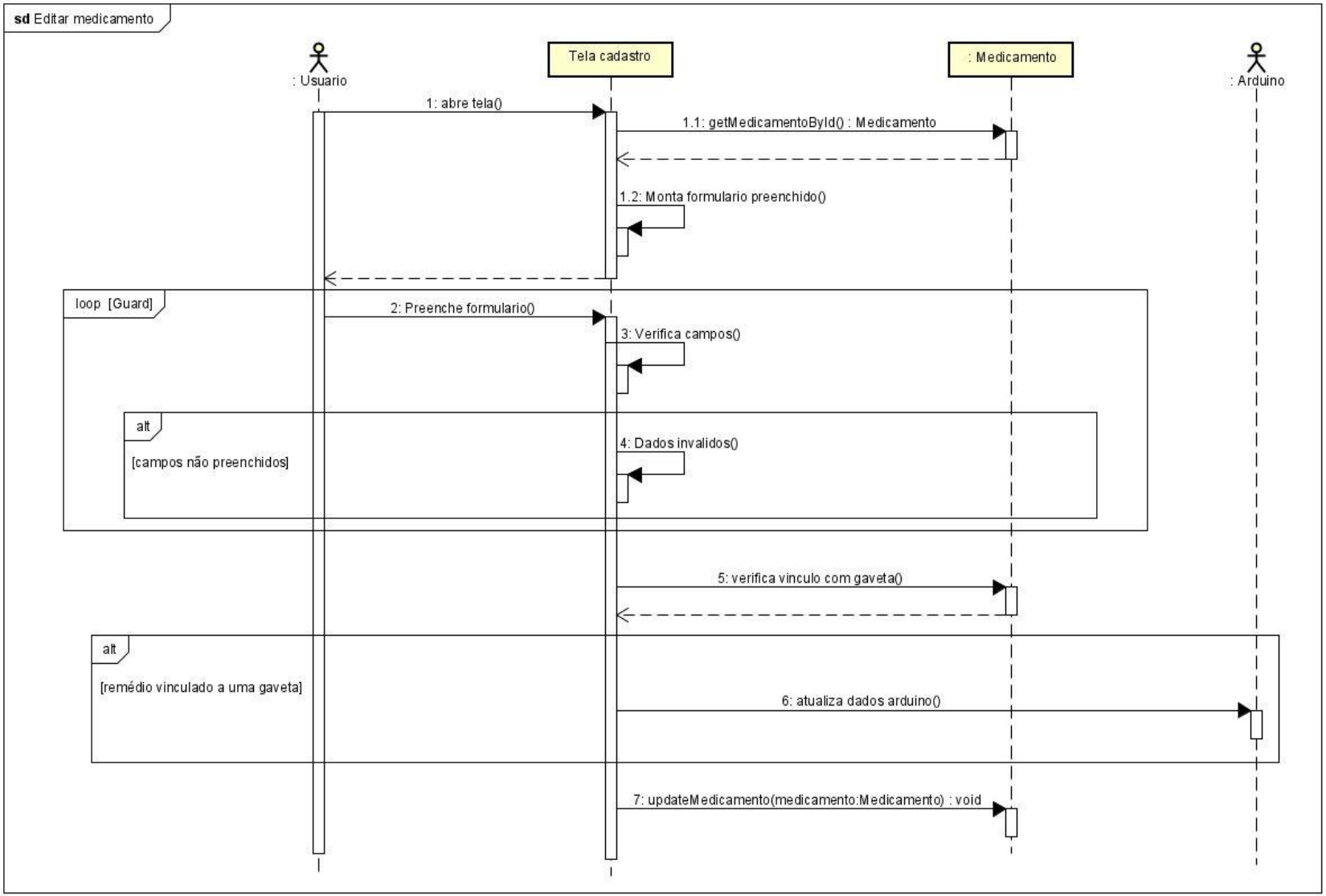
FONTE: OS AUTORES (2022).

FIGURA 40 - DIAGRAMA DE SEQUÊNCIA CADASTRAR MEDICAMENTO



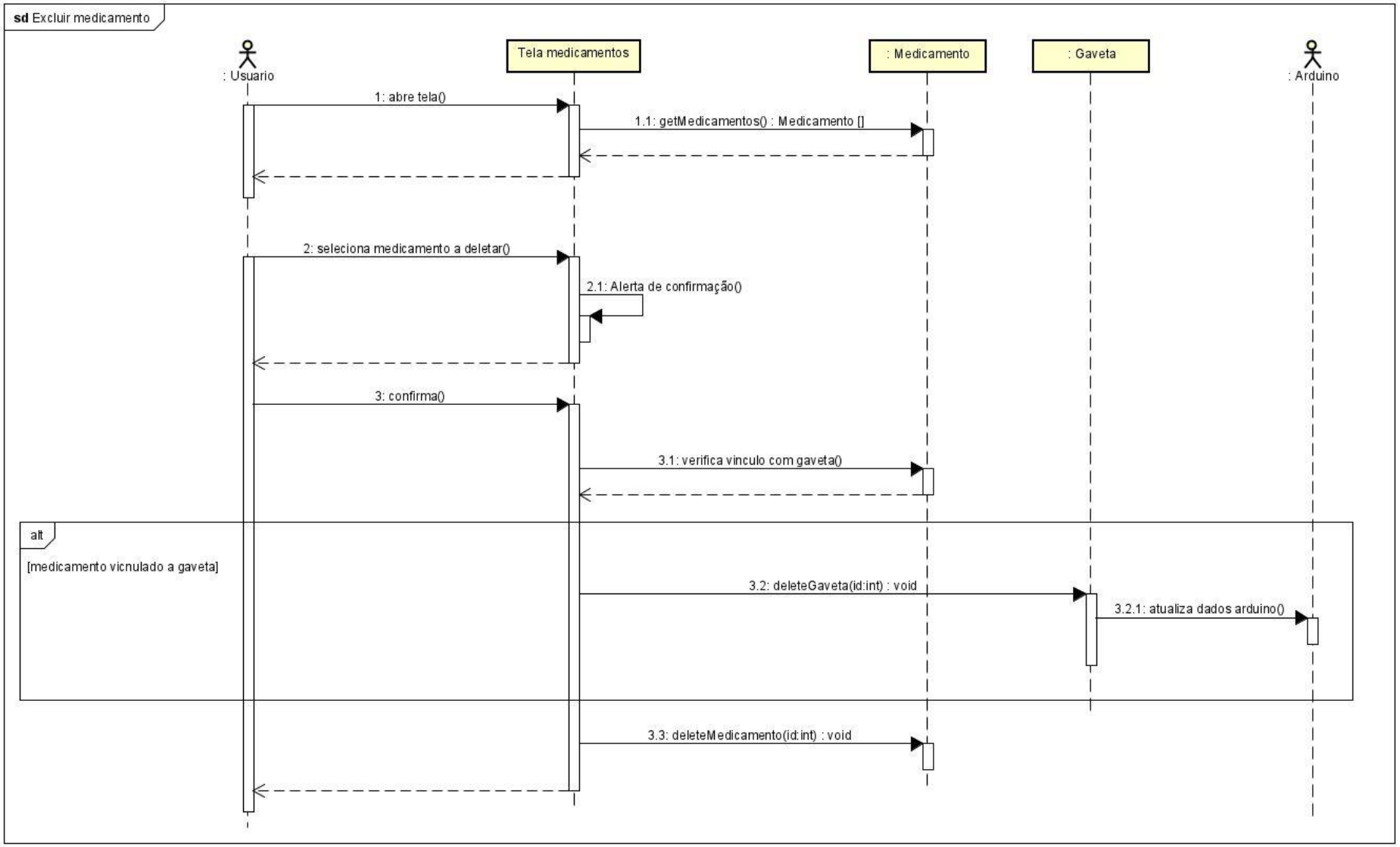
FONTE: OS AUTORES (2023).

FIGURA 41 - DIAGRAMA DE SEQUÊNCIA EDITAR MEDICAMENTO



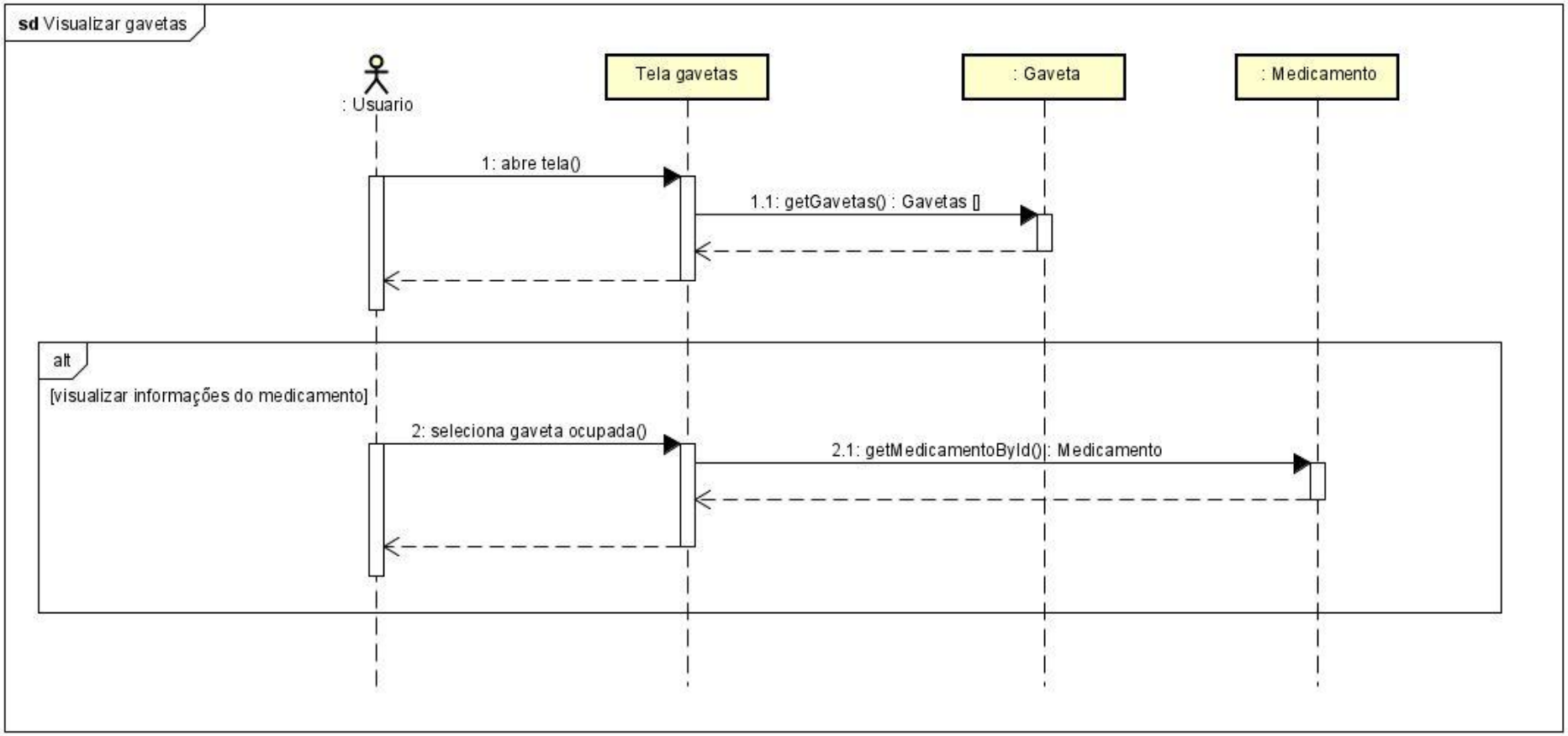
FONTE: OS AUTORES (2023).

FIGURA 42 - DIAGRAMA DE SEQUÊNCIA REMOVER MEDICAMENTO



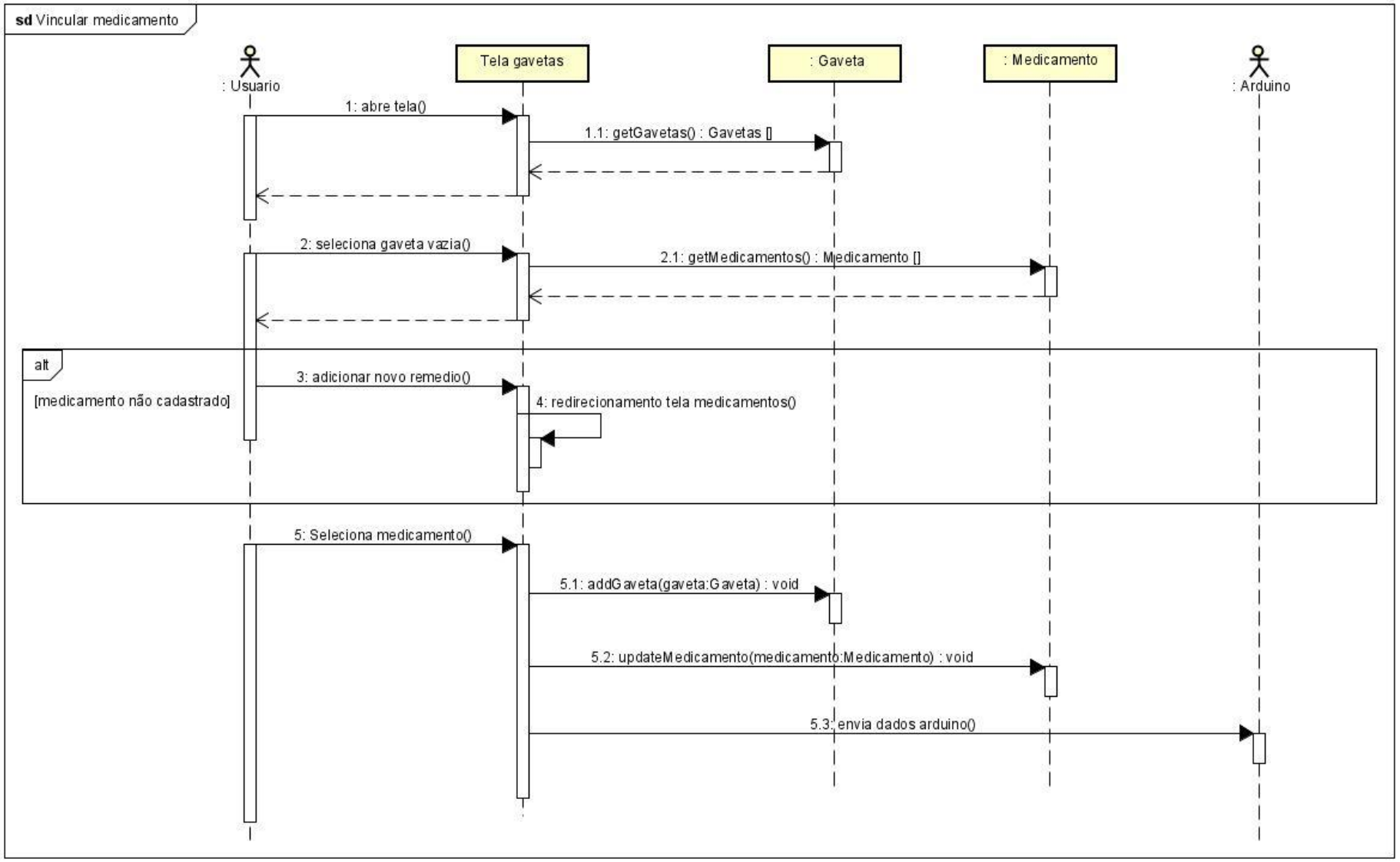
FONTE: OS AUTORES (2023).

FIGURA 43 - DIAGRAMA DE SEQUÊNCIA VISUALIZAR GAVETAS



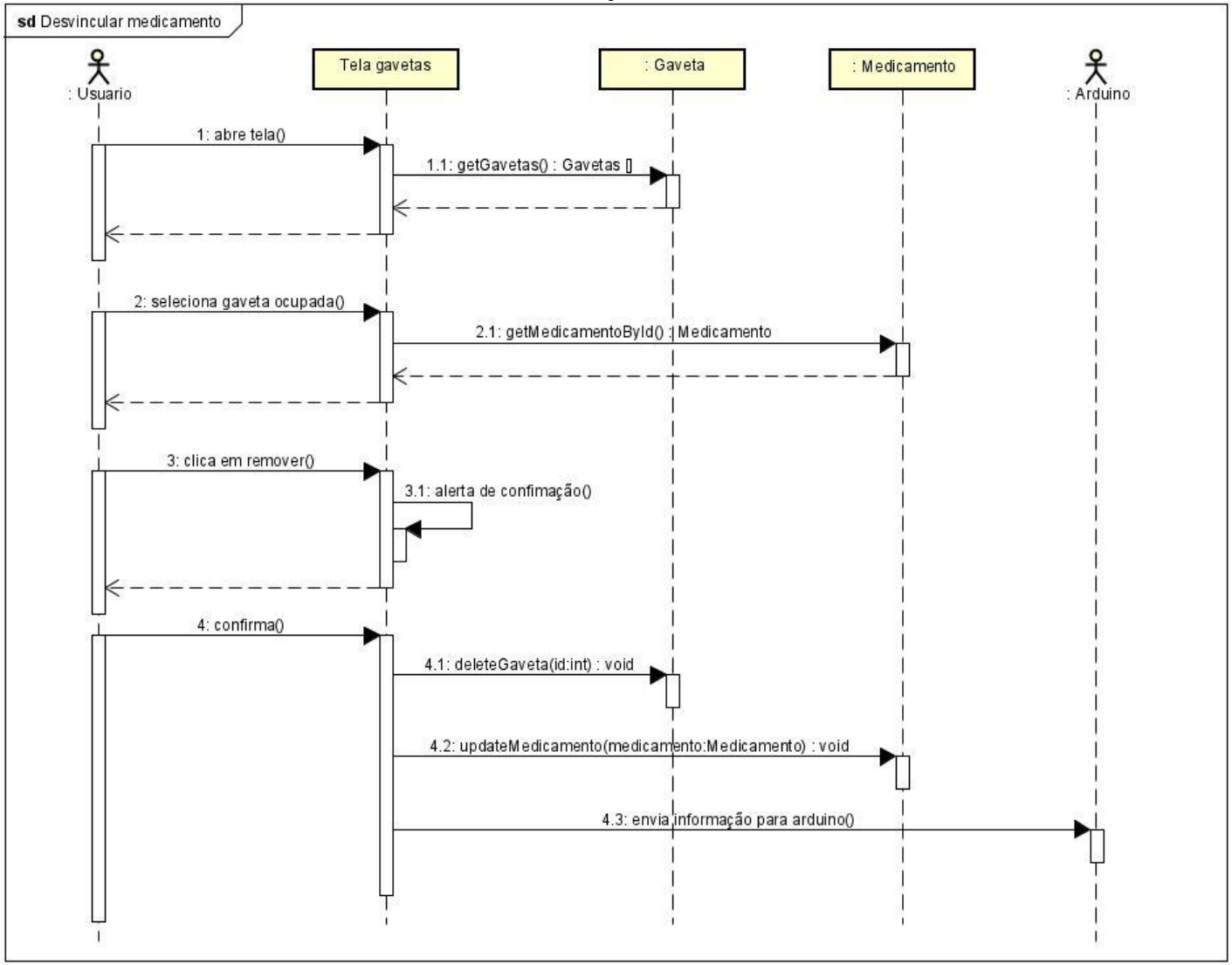
FONTE: OS AUTORES (2023).

FIGURA 44 - DIAGRAMA DE SEQUÊNCIA VINCULAR GAVETA



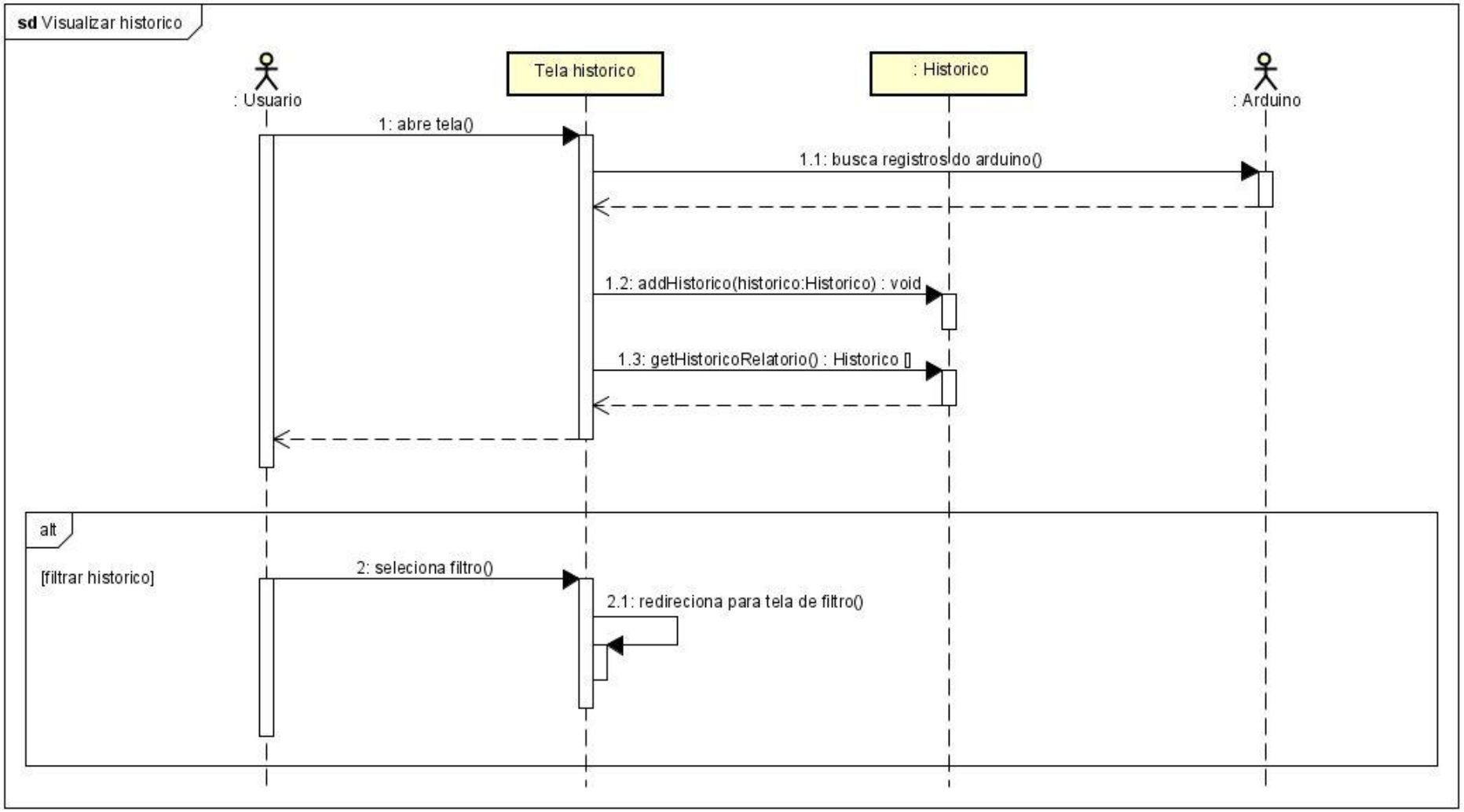
FONTE: OS AUTORES (2023).

FIGURA 45 - DIAGRAMA DE SEQUÊNCIA DESVINCULAR GAVETA



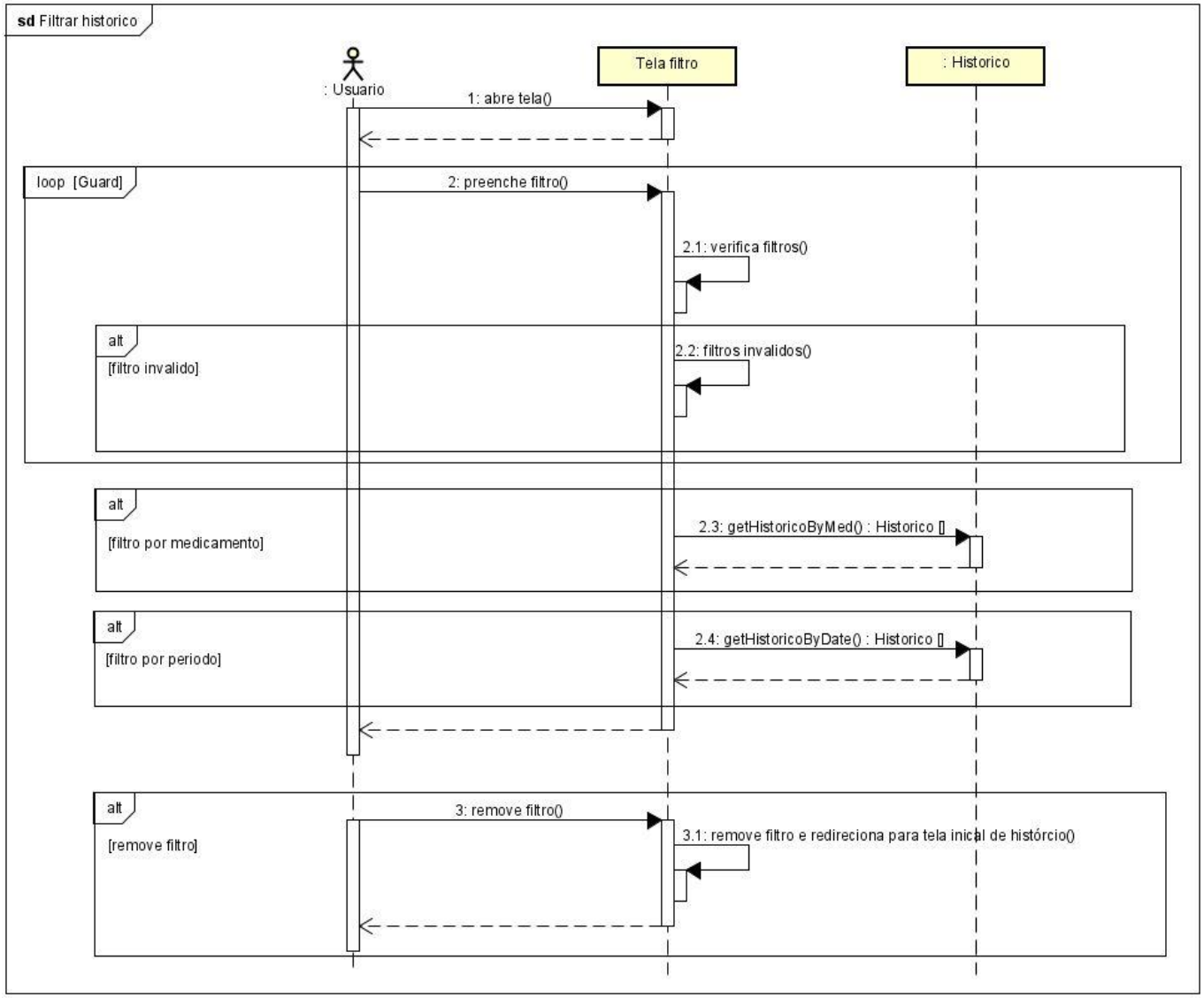
FONTE: OS AUTORES (2023).

FIGURA 46 - DIAGRAMA DE SEQUÊNCIA VISUALIZAR HISTÓRICO



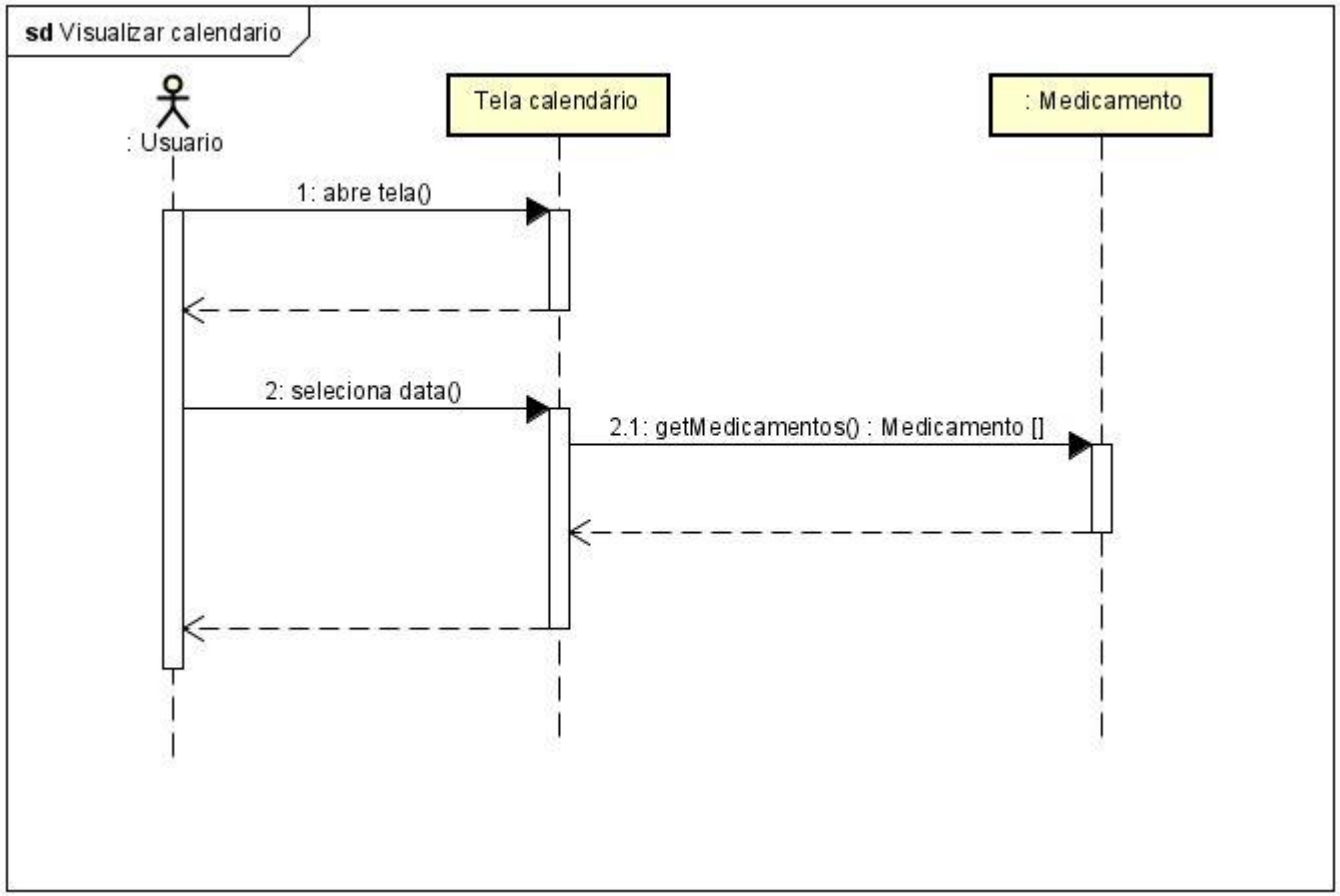
FONTE: OS AUTORES (2023).

FIGURA 47 - DIAGRAMA DE SEQUÊNCIA FILTRAR HISTÓRICO



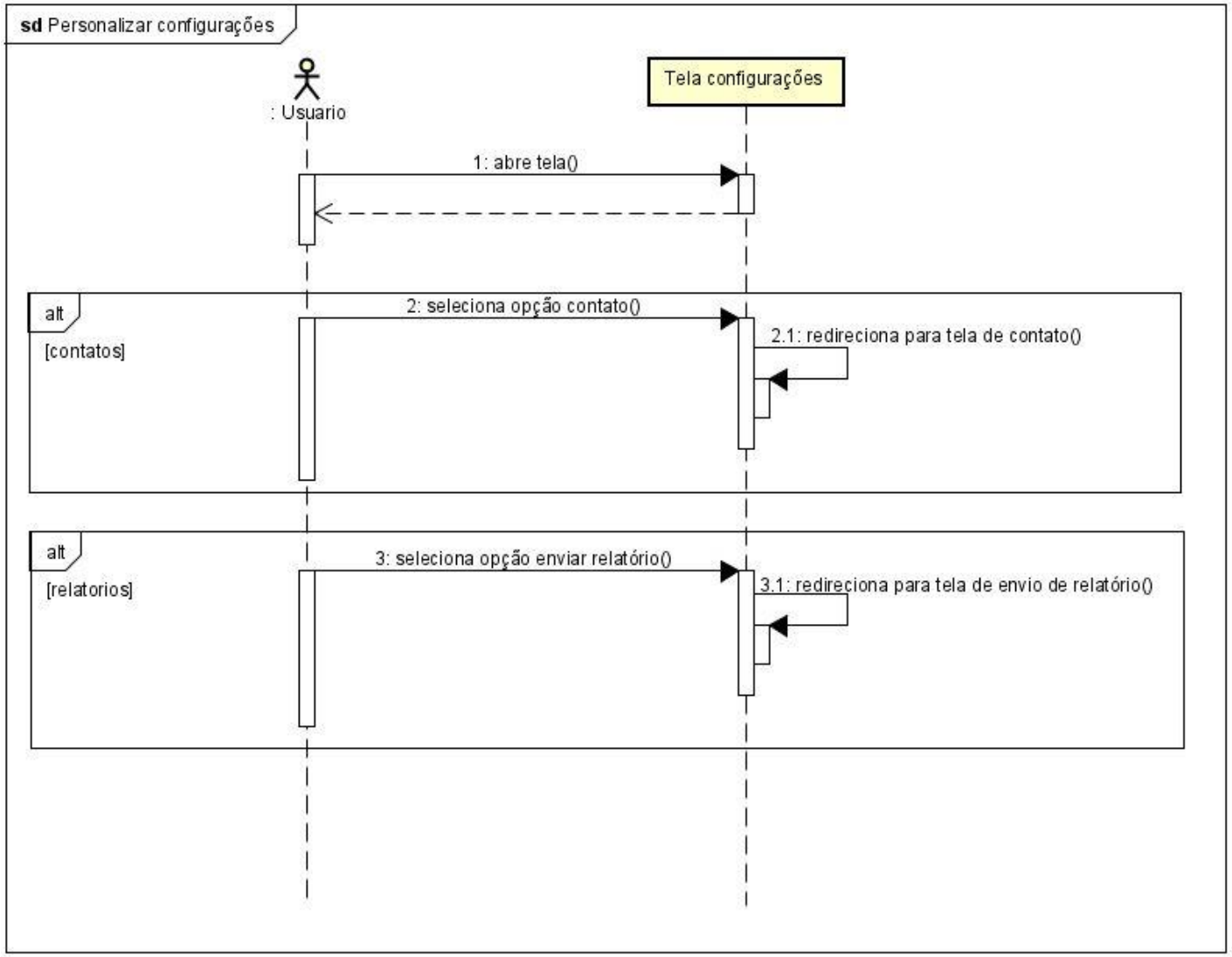
FONTE: OS AUTORES (2023).

FIGURA 48 - DIAGRAMA DE SEQUÊNCIA VISUALIZAR CALENDÁRIO



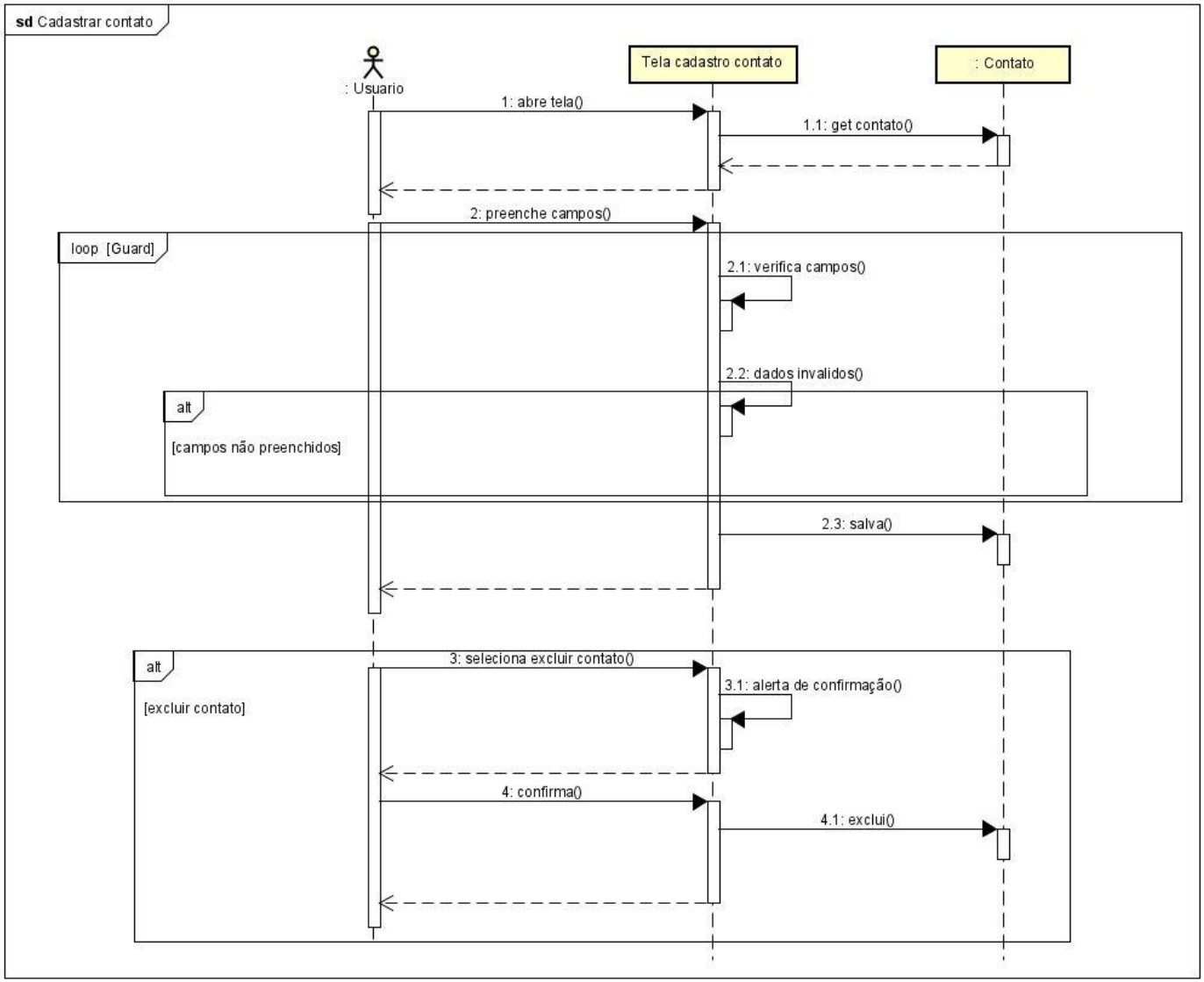
FONTE: OS AUTORES (2023).

FIGURA 49 - DIAGRAMA DE SEQUÊNCIA PERSONALIZAR CONFIGURAÇÕES



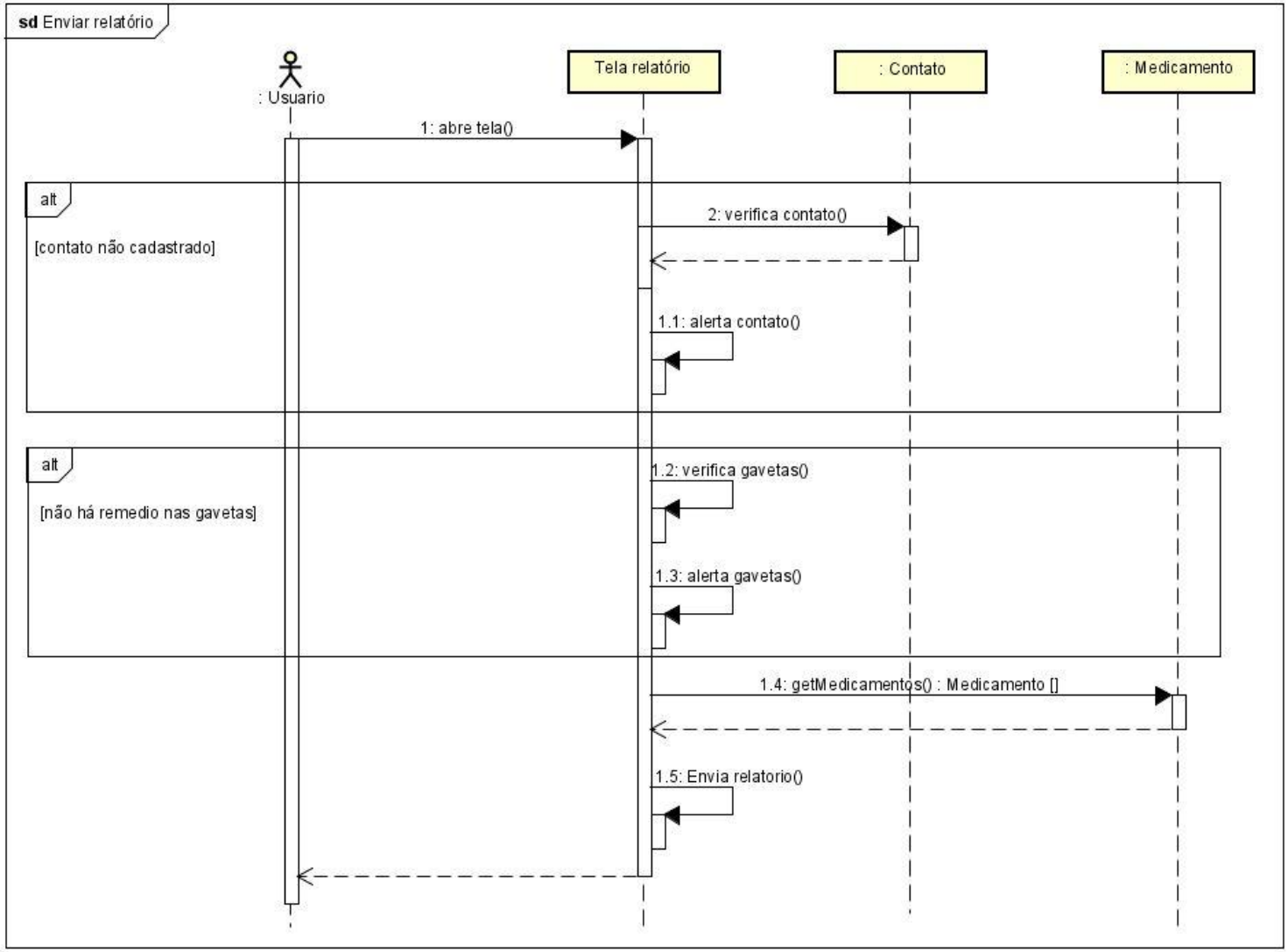
FONTE: OS AUTORES (2023).

FIGURA 50 - DIAGRAMA DE SEQUÊNCIA CADASTRAR CONTATO



FONTE: OS AUTORES (2023).

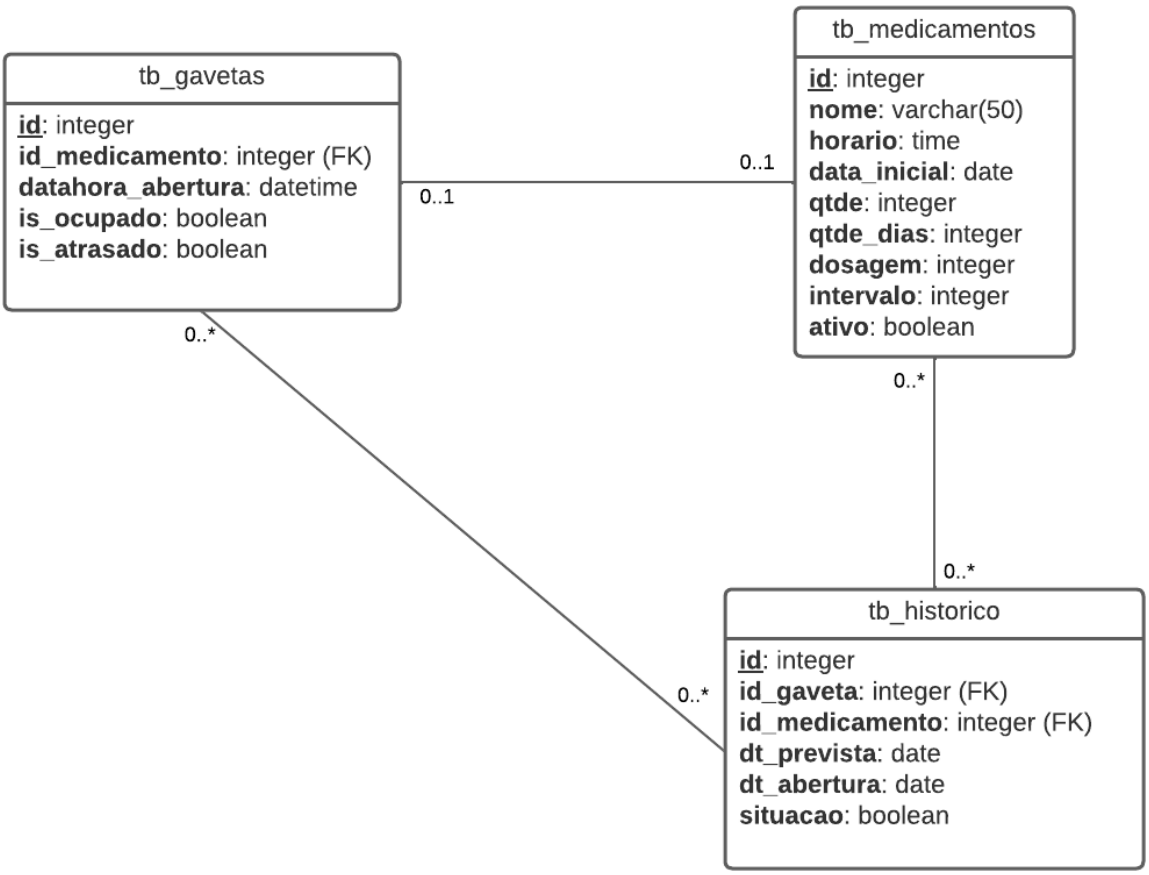
FIGURA 51 - DIAGRAMA DE SEQUÊNCIA ENVIAR RELATÓRIO



FONTE: OS AUTORES (2023).

APÊNDICE F – DIAGRAMA ENTIDADE RELACIONAMENTO

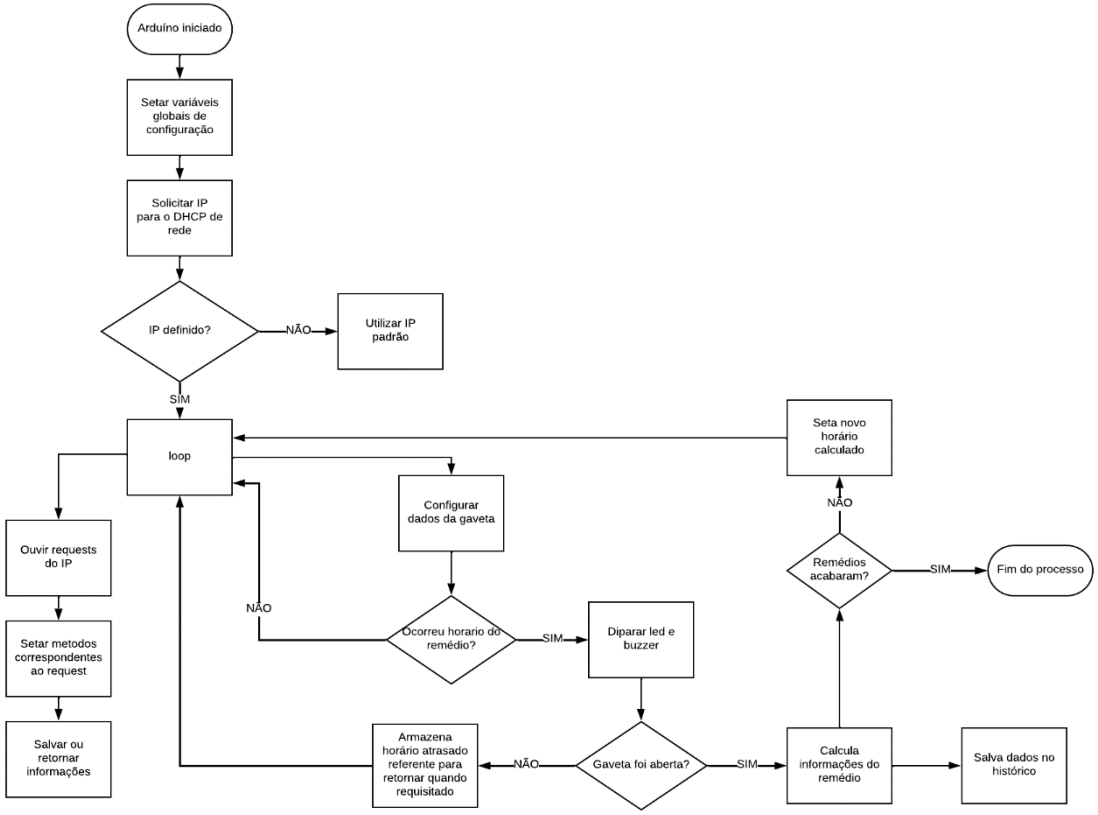
FIGURA 52 - DIAGRAMA ENTIDADE RELACIONAMENTO APLICATIVO



FONTE: OS AUTORES (2023).

APÊNDICE G – FLUXOGRAMA ARDUINO

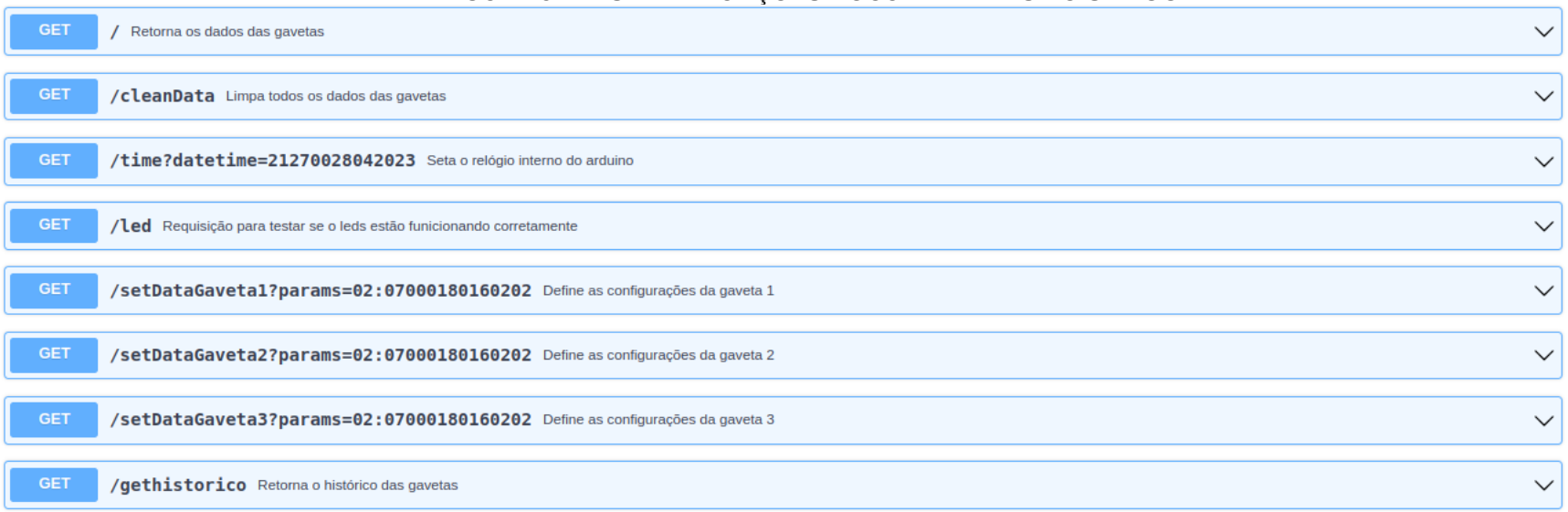
FIGURA 53 - FLUXOGRAMA ARDUINO



FONTE: OS AUTORES (2023).

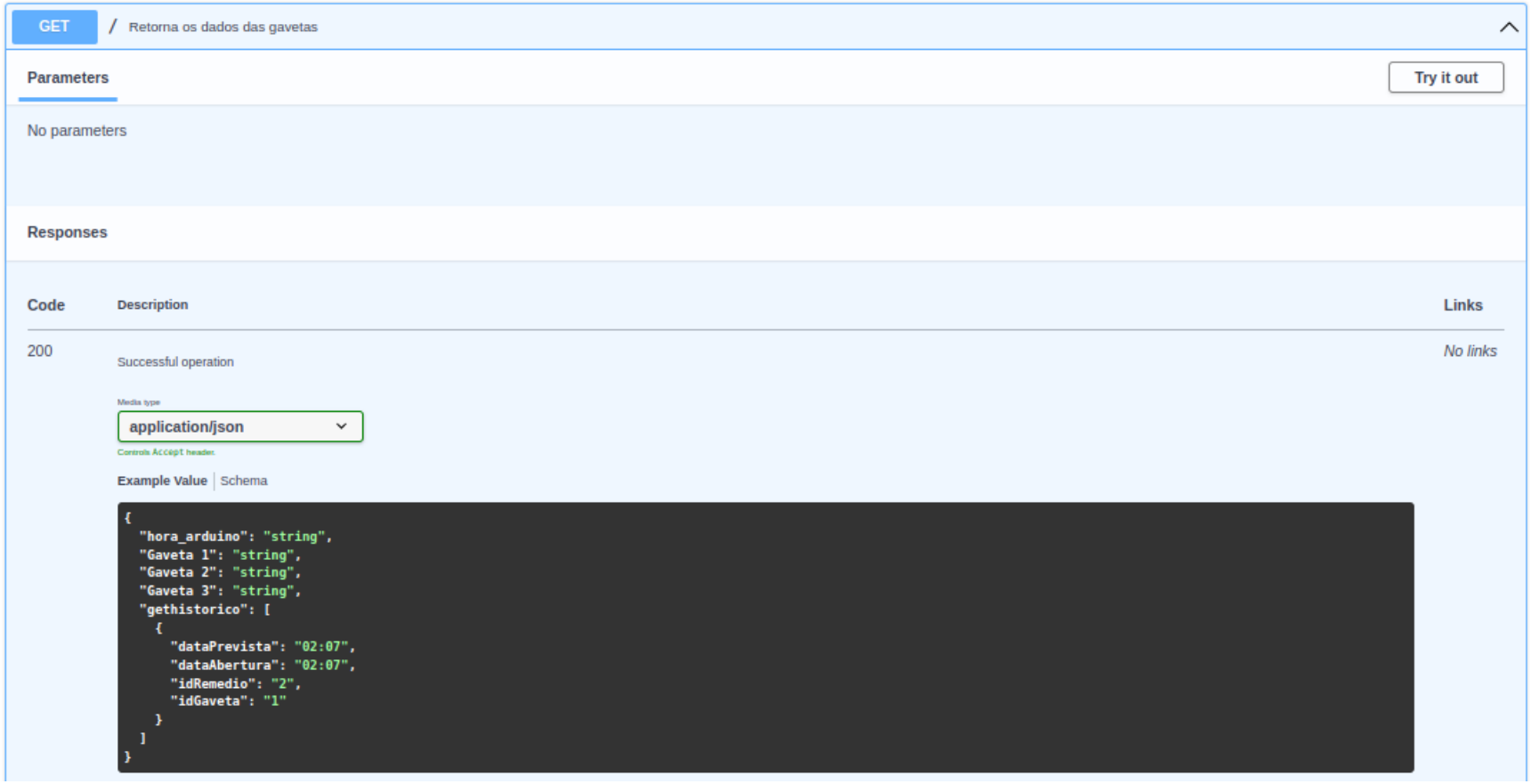
APÊNDICE H – ESPECIFICAÇÃO DA API DO ARDUINO

FIGURA 54 - LISTA DE FUNÇÕES DOCUMENTADAS NO SWAGGER



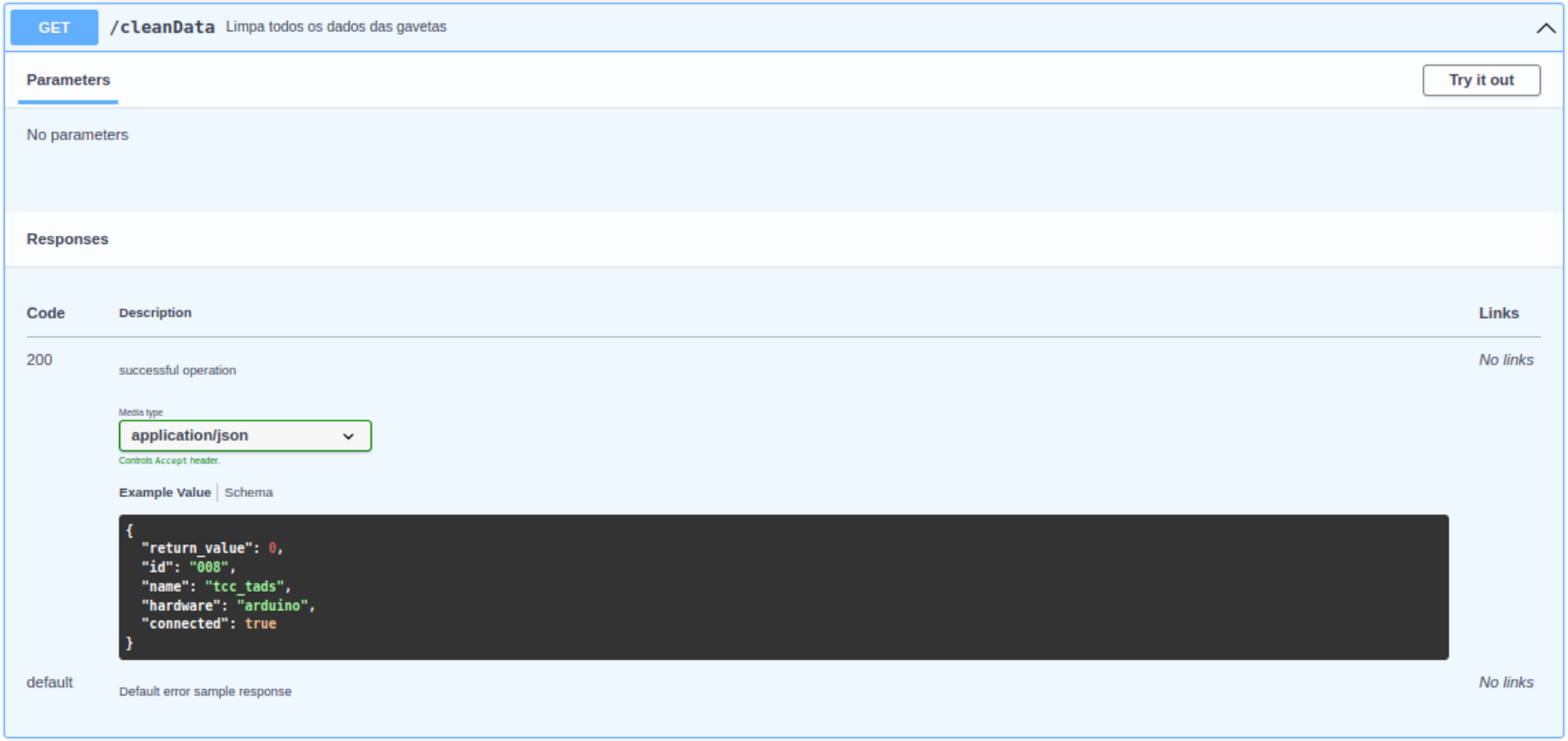
FONTE: OS AUTORES (2023).

FIGURA 55 - ROTA DA API PARA RETORNAR OS DADOS DAS GAVETAS



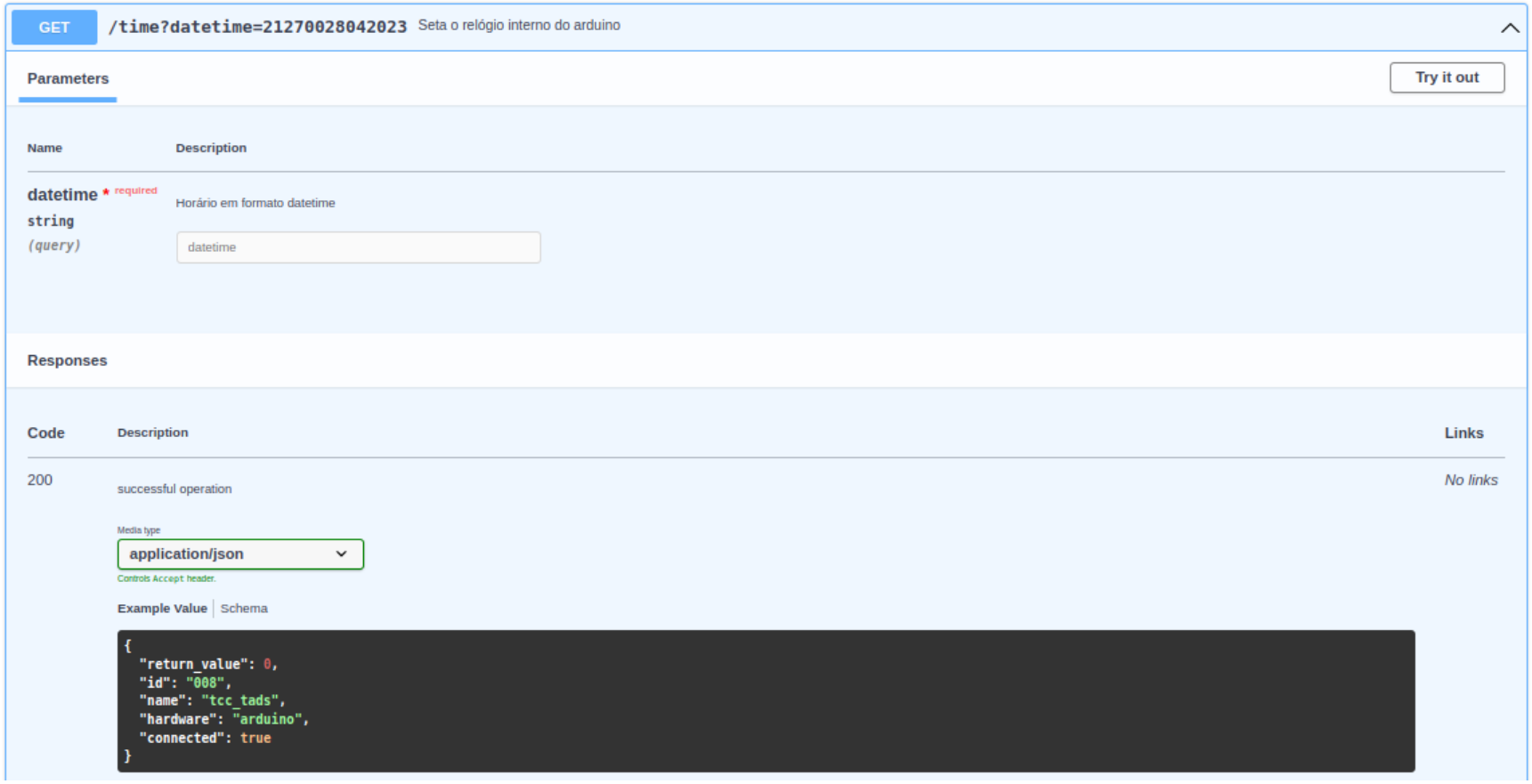
FONTE: OS AUTORES (2023).

FIGURA 56 - ROTA DA API PARA LIMPAR OS DADOS DOCUMENTADA NO SWAGGER



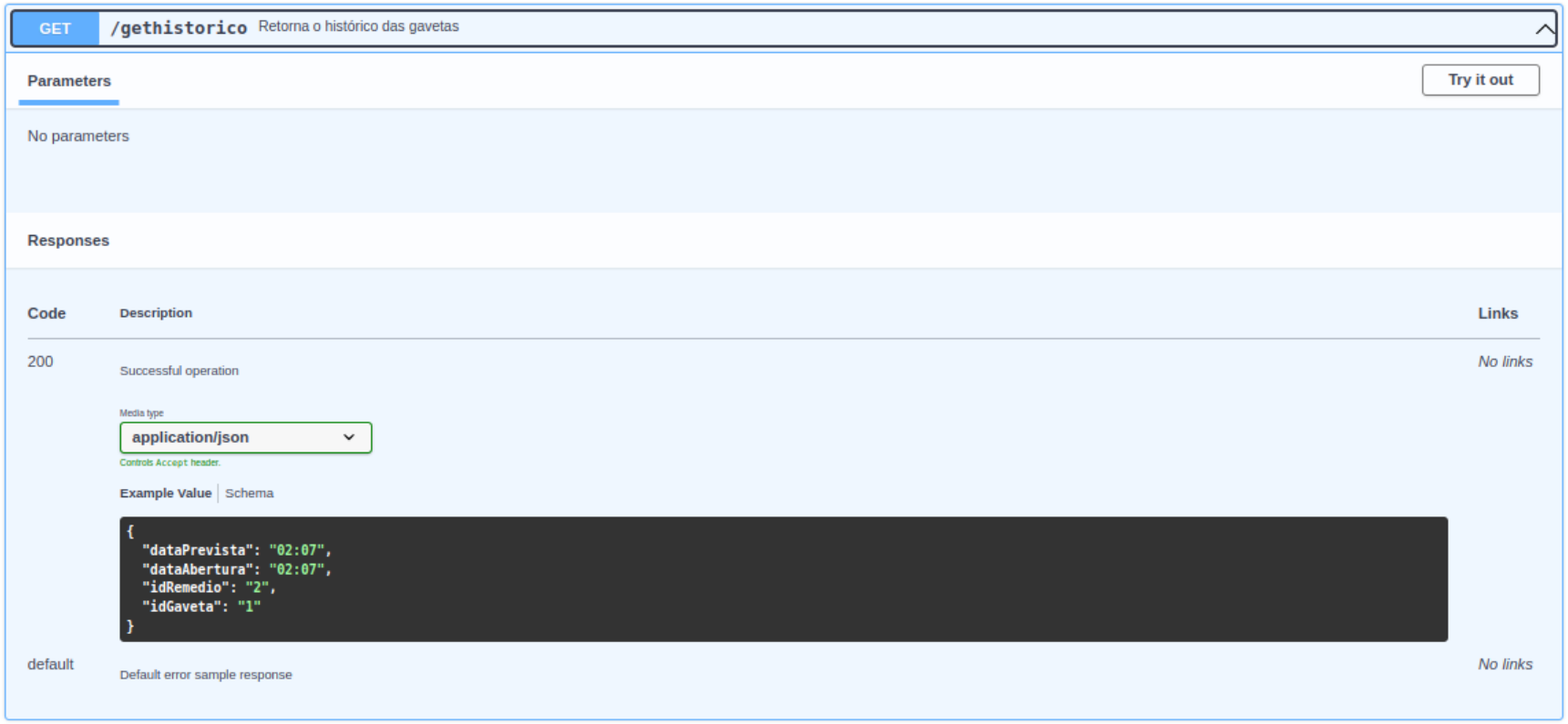
FONTE: OS AUTORES (2023).

FIGURA 57 - ROTA DA API PARA DEFINIR O RELÓGIO DO ARDUINO DOCUMENTADA NO SWAGGER



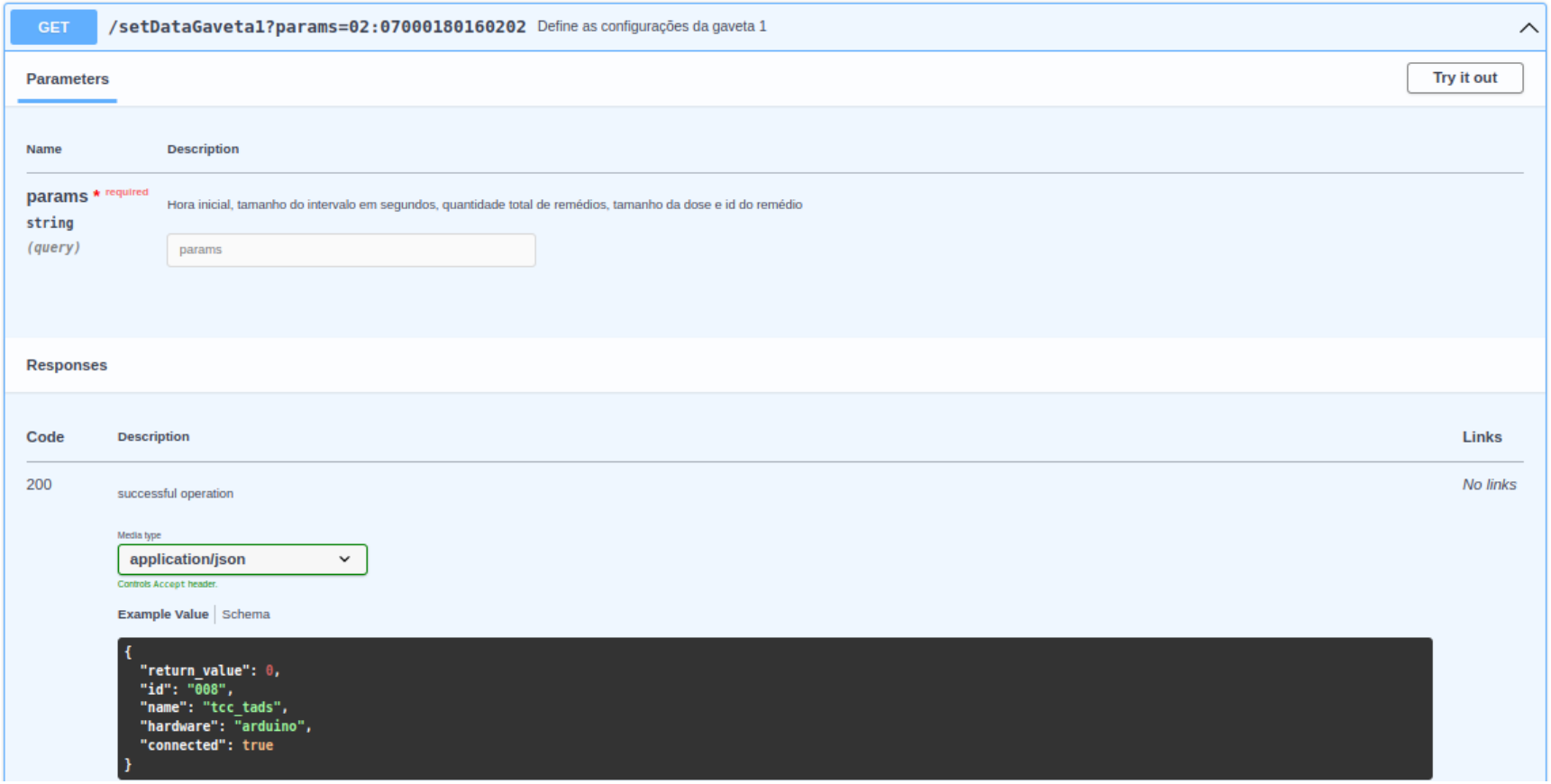
FONTE: OS AUTORES (2023).

FIGURA 58 - ROTA DA API PARA RETORNAR O HISTÓRICO DOCUMENTADO NO SWAGGER



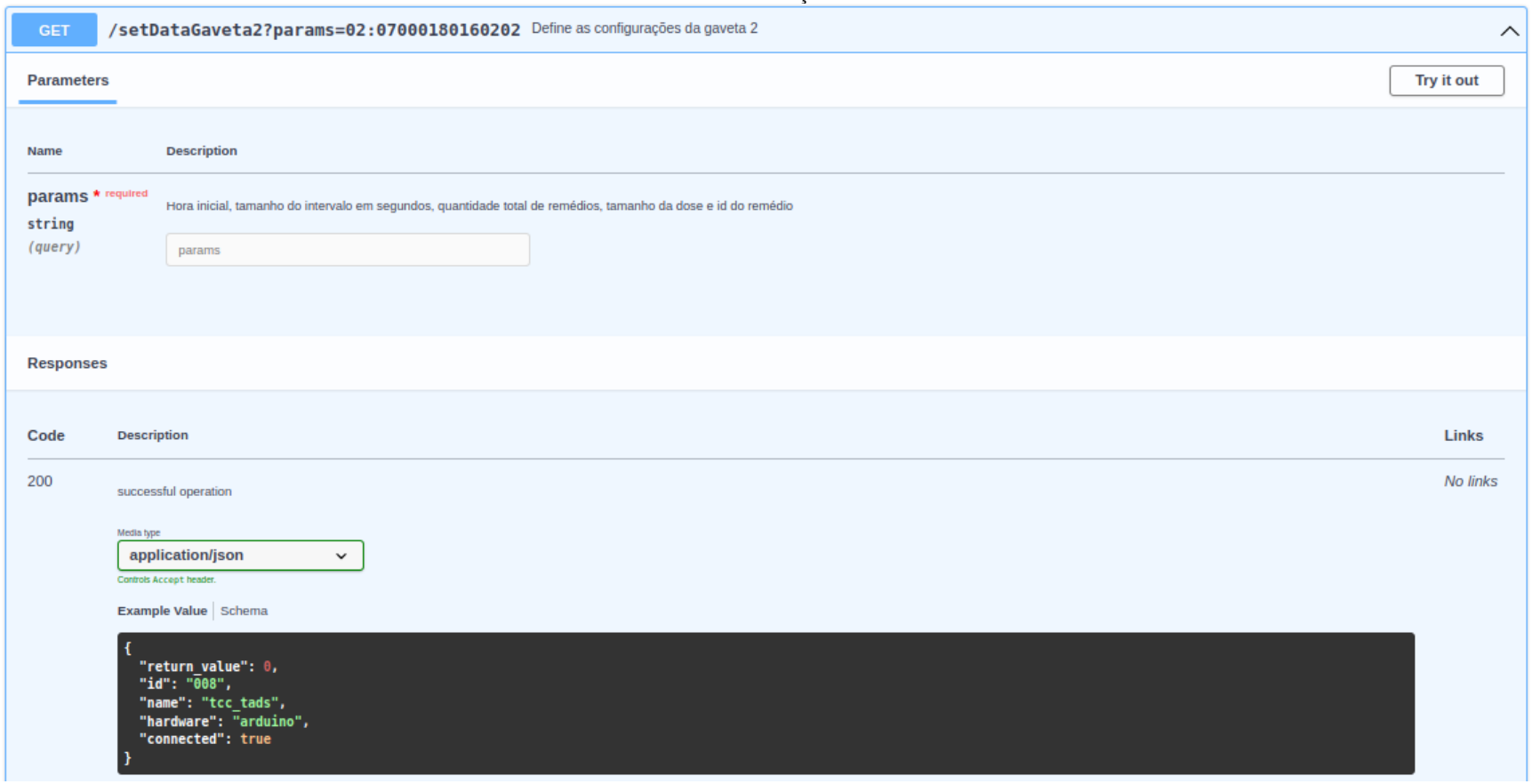
FONTE: OS AUTORES (2023).

FIGURA 59 - ROTA DA API PARA DEFINIR AS CONFIGURAÇÕES DA GAVETA 1 DOCUMENTADA NO SWAGGER



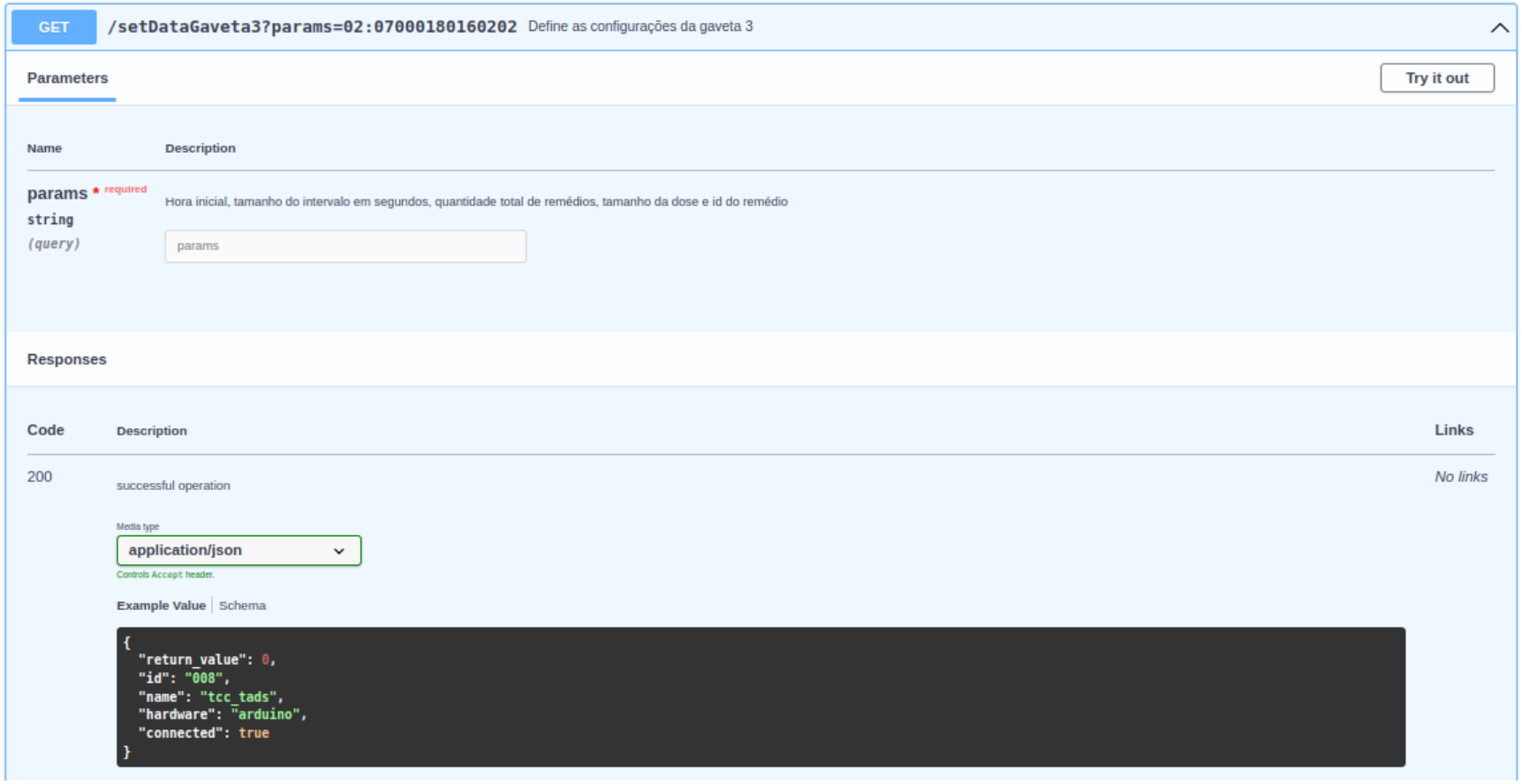
FONTE: OS AUTORES (2023).

FIGURA 60 - ROTA DA API PARA DEFINIR AS CONFIGURAÇÕES DA GAVETA 2 DOCUMENTADA NO SWAGGER



FONTE: OS AUTORES (2023).

FIGURA 61 - ROTA DA API PARA DEFINIR AS CONFIGURAÇÕES DA GAVETA 3 DOCUMENTADA NO SWAGGER



FONTE: OS AUTORES (2023).