Asimov: Um Laboratório Remoto para Ensino Experimental de Robótica

Gustavo Yoshio Maruyama[1,2], Amaury Antônio de Castro Junior[1], Anderson Corrêa de Lima[1], Marcos Pinheiro Vilhanueva[2]

[1] Faculdade de Computação - Universidade Federal de Mato Grosso do Sul (UFMS)
Av. Costa e Silva, s/nº - Bairro Universitário – Campo Grande, MS – Brasil

[2]Instituto Federal de Educação, Ciência e Tecnologia de Mato Grosso do Sul (IFMS)
Campus Coxim
Rua Salime Tanure, s/nº - Santa Tereza - Coxim, MS - Brasil

[3]Instituto Federal de Educação, Ciência e Tecnologia de Mato Grosso do Sul (IFMS)
Rodovia BR-463, Km 14, s/nº – Sanga Puitã - Ponta Porã, MS – Brasil

gustavo.maruyama@ifms.edu.br, {amaury.junior, anderson.lima}@ufms.br,

marcos.vilhanueva@ifms.edu.br

Abstract. *This article addresses the development, testing and evaluation of the Asimov - Remote Robotics Laboratory system. It is a remote experimentation environment that allows people, anywhere in the world, to carry out practical programming experiments on robots or microcontrolled devices remotely. The system was tested in practice through the application of a basic programming and robotics course entirely remotely in an asynchronous format, where the practical part took place through the developed system. Finally, a usability evaluation of the system was carried out based on the user's perception regarding usefulness, ease of use, graphical interface and teaching.*

Resumo. *Este artigo aborda o desenvolvimento, teste e avaliação do sistema do Asimov - Laboratório Remoto de Robótica. Trata-se de um ambiente de experimentação remota que permite que pessoas, em qualquer lugar do mundo, realizem experimentos práticos de programação em robôs ou dispositivos micro controlados a distância. O sistema foi testado na prática a partir da aplicação de um curso básico de programação e robótica inteiramente à distância em formato assíncrono, onde a parte prática ocorreu por meio do sistema desenvolvido. Por fim, foi realizado uma avaliação de usabilidade do sistema a partir da percepção do usuário em relação a utilidade, facilidade de uso, interface gráfica e ensino.*

1. Introdução

O uso da robótica na educação vem ganhando espaço em instituições de ensino fundamental, médio, técnico e superior. Diferentes artigos apontam que alunos se sentem mais motivados e engajados nos estudos, como relatado nos trabalhos de Vidal et al. (2021) e Dargains e Sampaio (2020). Já em Andriola (2021), são apontados os principais benefícios do uso da robótica na educação, também chamada de robótica pedagógica ou robótica educacional. Dentre os benefícios, destacam-se: estímulo do raciocínio lógico,

auxílio na organização mental, incentivo ao aprendizado de matemática, física e língua inglesa, estímulo à criatividade e ao desenvolvimento de habilidades para solucionar problemas.

Apesar dos diversos fatores favoráveis, existem problemas que dificultam a adoção da robótica educacional nas instituições de ensino, tais como a indisponibilidade de tempo para planejamento e realização de atividades de robótica, formação teórico-prática do professores e custos relacionados a equipamentos que impactam diretamente na montagem de um laboratório específico [Campos 2017]. No Brasil, por exemplo, muitas escolas públicas carecem de recursos para implantação de um laboratório robótica.

Por outro lado, no caso das instituições de ensino que possuem laboratórios específicos de robótica, podem surgir situações que impeçam o acesso às dependências físicas da instituição. Por exemplo, no ano de 2020 o mundo passou a enfrentar uma grave crise sanitária, a pandemia de COVID-19 (doença do coronavírus) [Bavel et al. 2020]. No Brasil, o primeiro caso foi relatado em 26 de fevereiro de 2020, desde então a doença se espalhou rapidamente em todo território nacional, causando impactos sociais e econômicos [Souza et al. 2020]. Como medida de contenção ao vírus, instituições de ensino suspenderam o acesso às suas dependências físicas, tornando ociosos as suas salas de aula e laboratórios de ensino.

Este artigo aborda o desenvolvimento, teste e avaliação de um sistema de laboratório remoto de robótica denominado Asimov - Laboratório Remoto de Robótica. Trata-se de um ambiente de experimentação remota que permite que pessoas, em qualquer lugar do mundo, realizem experimentos práticos de programação em robôs ou dispositivos microcontrolados compartilhados por instituições de ensino que possuam um laboratório físico de robótica. Este sistema é uma alternativa para situações de suspensão ou impedimento do acesso presencial às dependências físicas de uma instituição de ensino. Além disso, possibilita o compartilhamento dos recursos com instituições de ensino que não possuem laboratórios específicos de robótica.

Para obtenção dos resultados, foi aplicado um curso à distância de programação e robótica em formato assíncrono, onde a parte prática ocorreu por meio do sistema desenvolvido. Neste curso foram abordados conceitos básicos de programação e técnicas para capacitar os participantes a programar um robô seguidor de linha. Durante o curso, os participantes preencheram um formulário de avaliação do sistema.

2. Laboratório Remoto X Laboratório Virtual

Um laboratório remoto é um sistema composto por *hardware* e *software*, permitindo que o usuário manipule, a distância, os equipamentos de um laboratório físico para realização de experimentos práticos [Orduna et al. 2016]. Em um laboratório remoto, o usuário pode ter um *feedback* visual do experimento através de imagens capturadas por câmeras, ou seja, é possível acompanhar visualmente e em tempo real, todo o experimento realizado. O artigo de Almeida et al. (2017) traz um exemplo de um laboratório remoto de robótica que pode ser acessado por meio de um navegador web através da internet. Neste laboratório, o usuário pode escrever e submeter um código que será programado em um robô construído com o kit robótico Lego Mindstorm. No próprio navegador, o usuário pode visualizar o robô realizando as ações por meio de uma câmera instalada no laboratório físico. Já em um laboratório virtual, existe um sistema que simula a infraestru-

tura de um laboratório real por meio de *softwares* [Flores et al. 2018], conhecidos como simuladores. Os laboratórios virtuais permitem que experimentos sejam realizados em diversas condições, sem a preocupação de danos aos equipamentos e sem oferecer riscos aos usuários [Gomes and Bogosyan 2009] [Lin et al. 2020]. Nessas situações, é possível perceber a redução de custos de implantação e manutenção, quando comparados a um laboratório físico [Nedic et al. 2003]. No artigo de Kerem Erdoğmuş e Yayan (2021), são apresentados exemplos de laboratórios virtuais de robótica, com destaque ao EvaMars, um laboratório que simula um robô de treinamento em Marte. Esse é um exemplo onde o laboratório virtual se torna muito útil, pois manipular um robô real em Marte ainda é uma situação inviável à estudantes.

3. Revisão Sistemática da Literatura

Na Revisão Sistemática da Literatura (RSL) realizada por Maruyama et al. (2022), são indicadas motivações para utilização de laboratórios remotos. De acordo com Di Giamberardino e Temperini (2017), artigo estudado na RSL, a interação com processos físicos reais proporcionada por um laboratório remoto, pode ser uma característica mais atraente para o estudante do que um processo simulado em um laboratório virtual. Porém, vale ressaltar que a adoção de um laboratório remoto ou um laboratório virtual depende das necessidades e situações físicas e econômicas da instituição de ensino. Em outro artigo abordado na RSL, Nedic et al. (2003) sugere que laboratórios remotos e virtuais podem ser utilizados em conjunto, de forma complementar.

Instituições de ensino que possuem laboratórios físicos de robótica podem optar pela implantação de um laboratório remoto, justamente para fazer o melhor aproveitamento dos equipamentos disponíveis e dos investimentos já realizados. Dessa forma, os alunos e professores da instituição poderão utilizar o laboratório, mesmo em situações adversas que impeçam o acesso presencial, reduzindo assim, a ociosidade do laboratório físico. Segundo Gomes e Bogosyan (2009), também é possível compartilhar o laboratório com instituições de ensino municipais, estaduais e particulares que não possuem laboratório de robótica, ampliando a colaboração em ações educacionais entre as instituições. Neste contexto, algumas ações podem ser realizadas pela a instituição anfitriã do laboratório, como cursos, capacitações, treinamentos e oficinas a distância.

Além das motivações, a RSL responde questões que ajudaram na decisão de modelo de arquitetura, protocolos, linguagens, plataformas e ferramentas empregadas no desenvolvimento deste sistema (detalhes na seção 4).

4. Desenvolvimento do Sistema

O sistema Asimov - Laboratório Remoto de Robótica foi projetado conforme o esquema apresentado na Figura 1. Este esquema foi baseado diretamente no modelo de arquitetura abstraído de trabalhos correlatos abordados na revisão sistemática da literatura de Maruyama et al. (2022).

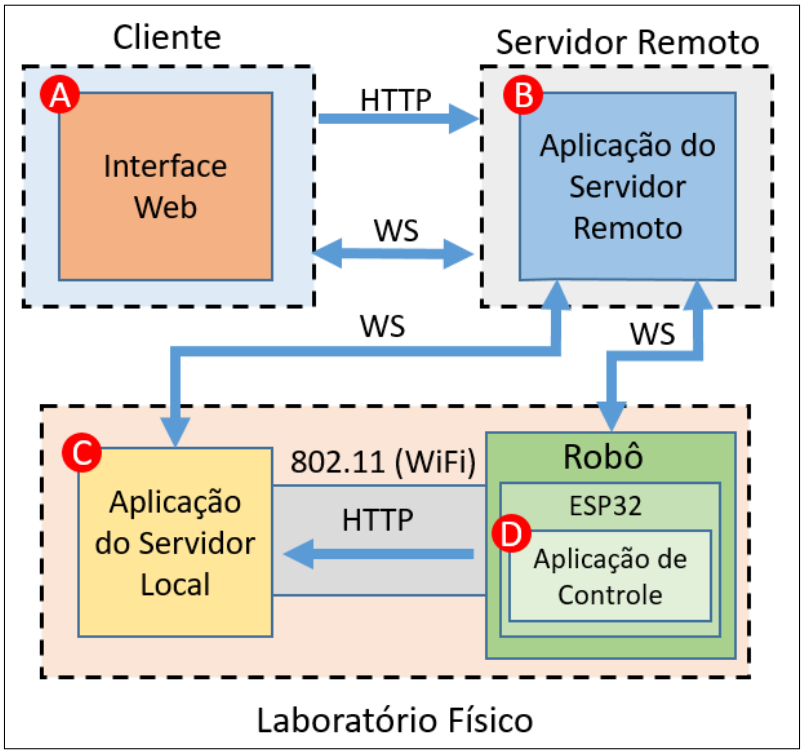

Figura 1. Esquema da arquitetura do sistema desenvolvido. Fonte: Autoria própria.

O sistema é composto por quatro aplicações, sendo:

1. Interface web (Figura 1-**A**), que é a aplicação onde o usuário interage diretamente.
2. Aplicação do servidor remoto (Figura 1-B), que basicamente faz a intermediação entre a interface web e as aplicações presentes no laboratório físico, principalmente por meio do protocolo *websocket* (WS).
3. Aplicação do servidor local (Figura 1-**C**), que se encontra no laboratório físico e é responsável por realizar a compilação do código fonte do usuário.
4. Aplicação de controle, responsável por realizar a rotina de atualização *Over-the-Air* (OTA) no ESP32 (Figura 1-**D**).

O sistema permite a escrita de código fonte diretamente em uma interface *web*, o envio do código para ser compilado em um computador no laboratório físico e o carregamento do programa em uma placa de desenvolvimento composto por um microcontrolador. O usuário pode visualizar o experimento em tempo real por meio de uma *webcam* instalada no laboratório. O sistema ainda fornece um monitor serial para que o usuário interaja com a interface de comunicação serial do dispositivo e também um *joystick* virtual programável, que dispara comandos diretamente na porta serial do dispositivo.

A placa de desenvolvimento escolhida para ser embarcada em robôs ou experimentos microcontrolados do sistema, é a placa ESP32 DEVKIT V1 (Fig. 2), esta placa possui um microcontrolador modelo ESP32-WROOM32 fabricado pela empresa Espressif. Esta placa foi escolhida para este projeto por quatro motivos:

1. Os microcontroladores da família ESP32 possuem conectividade sem fio padrão 802.11 b/g/n (popularmente conhecido por *WiFi*) e *Bluetooth* v4.2 BLE (*Bluetooth*

Low Energy) [Espressif 2022b]. A conectividade sem fio é importante para este projeto por permitir que o microcontrolador seja programado sem estar conectado fisicamente por um cabo USB ao computador do laboratório. A ausência de cabos também beneficia a movimentação do robô.

2. A Espressif fornece o *framework* denominado ESP-IDF (*Espressif IoT Development Framework*), que facilita a utilização dos recursos dos microcontroladores ESP32. Está presente neste *framework*, o mecanismo de atualização OTA (*Over The Air Update*), que permite que seja carregado um novo *software* no microcontrolador por meio de conectividade sem fio, mesmo em tempo de execução [Espressif 2022a].
3. Os microcontroladores da família ESP32 são compatíveis com as bibliotecas do Arduino. Como abordado na revisão sistemática da literatura de Maruyama et al. (2022), o Arduino é a plataforma mais utilizada dentre os trabalhos analisados.
4. A placa ESP32 DEVKIT V1 tem um custo menor se comparado ao Arduino MKR1000 WIFI, que é o modelo de placa do Arduino que possui conectividade sem fio.

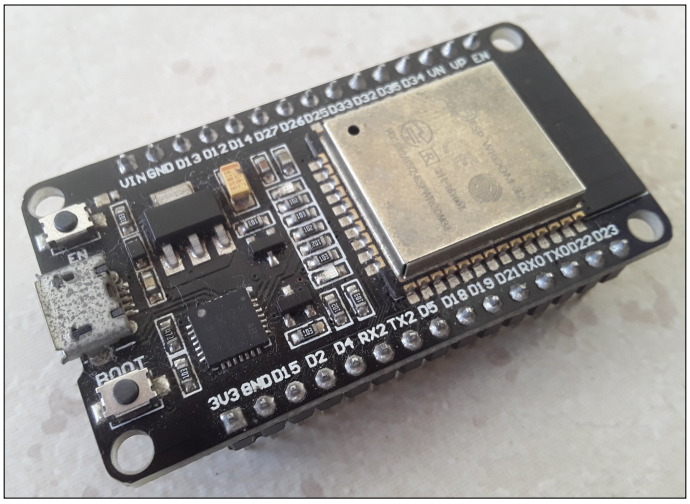

Figura 2. Placa de desenvolvimento ESP32 DEVKIT V1. Fonte: Autoria própria.

4.1. Upload, Compilação e Execução do Programa

Para que o usuário possa utilizar o robô ou experimento disponibilizado no laboratório remoto, é necessário que o mesmo crie um código usando a Linguagem C/C++ e o envio para o ESP 32. Nesta subseção é explicado o processo da principal funcionalidade do sistema, que inclui o *upload* do código fonte, compilação e execução do programa. Este processo é ilustrado por meio de um diagrama de sequência apresentado na Figura 3. Segue a explicação de cada mensagem do diagrama de sequência:

Mensagem 1: O usuário escreve um código fonte por meio do editor presente na interface web e depois realiza o envio do código para a aplicação no servidor remoto.
Mensagem 2: O servidor remoto envia, por meio de *websocket*, o código fonte para a aplicação do servidor local do laboratório físico para a compilação.
Mensagem 3: Após a compilação, a aplicação do servidor local do laboratório físico avisa o servidor remoto que a compilação terminou.

Mensagem 4: O servidor remoto envia uma mensagem para o microcontrolador ESP32 informando que o novo *software* está disponível.
Mensagem 5: O microcontrolador ESP32 realiza o *download* do novo programa.
Mensagem 6 e 7: O microcontrolador então atualiza o seu programa e reinicia.

Durante a compilação do código fonte, a aplicação do servidor local do laboratório físico envia mensagens da saída do console (contendo detalhes da compilação) para o servidor remoto, que as repassa para interface *web* para serem mostradas ao usuário.

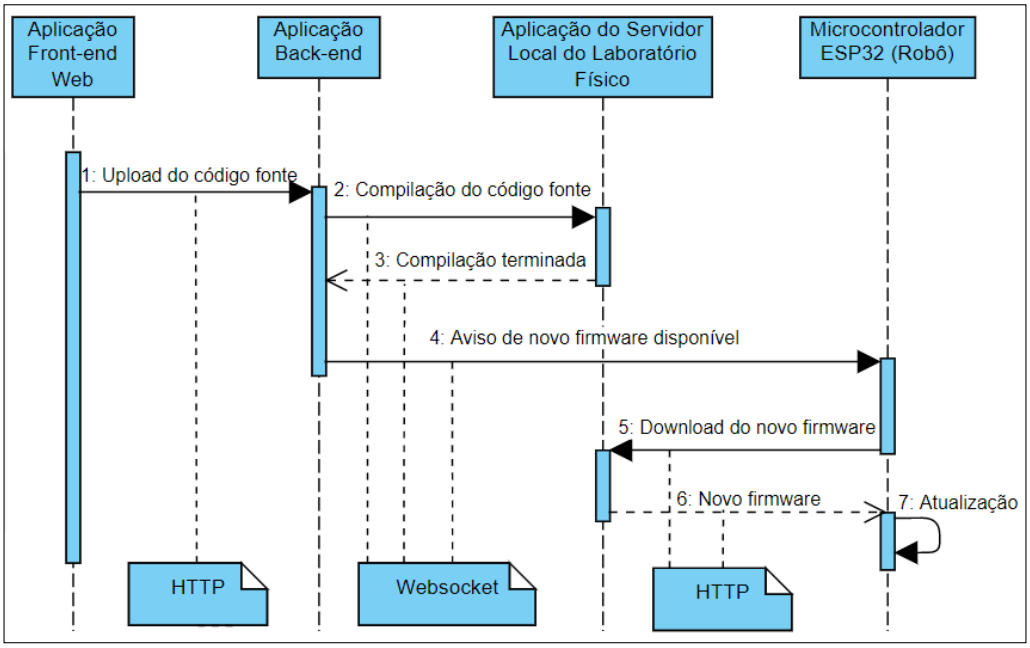

Figura 3. Diagrama de sequência. Fonte: Autoria própria.

4.2. Feedback Visual

O sistema foi desenvolvido para permitir a visualização em tempo real do experimento por meio de *webcams* comuns. Para que isso fosse possível, foi utilizado o protocolo WebRTC (*Web Real Time Communication*) na aplicação do servidor local. A aplicação possui uma interface *web* para compartilhamento da câmera, como pode ser visto na Figura 4. O protocolo WebRTC permite a conexão ponto a ponto entre navegadores *web* para a transmissão de dados oriundos de câmeras e microfones com o mínimo de latência possível [Alvestrand 2021].

Uma dificuldade encontrada na utilização do WebRTC foi a impossibilidade de conexão ponto a ponto entre clientes que estão localizados em redes atrás de roteadores com NAT (*Network Address Translation*) habilitado. Para contornar este problema, foi utilizado um servidor que funciona como um intermediador comunicação entre os clientes, retransmitindo os pacotes de dados entre ambos. Este servidor é denominado TURN (*Traversal Using Relays around NAT*) [Matthews et al. 2010]. Para este projeto foi utilizado uma implementação de servidor TURN *open source* denominada coTURN [Coturn 2024]. Tanto a aplicação do servidor remoto quanto o servidor TURN foram implantados em serviço PaaS (*Platform as a Service*) privado.

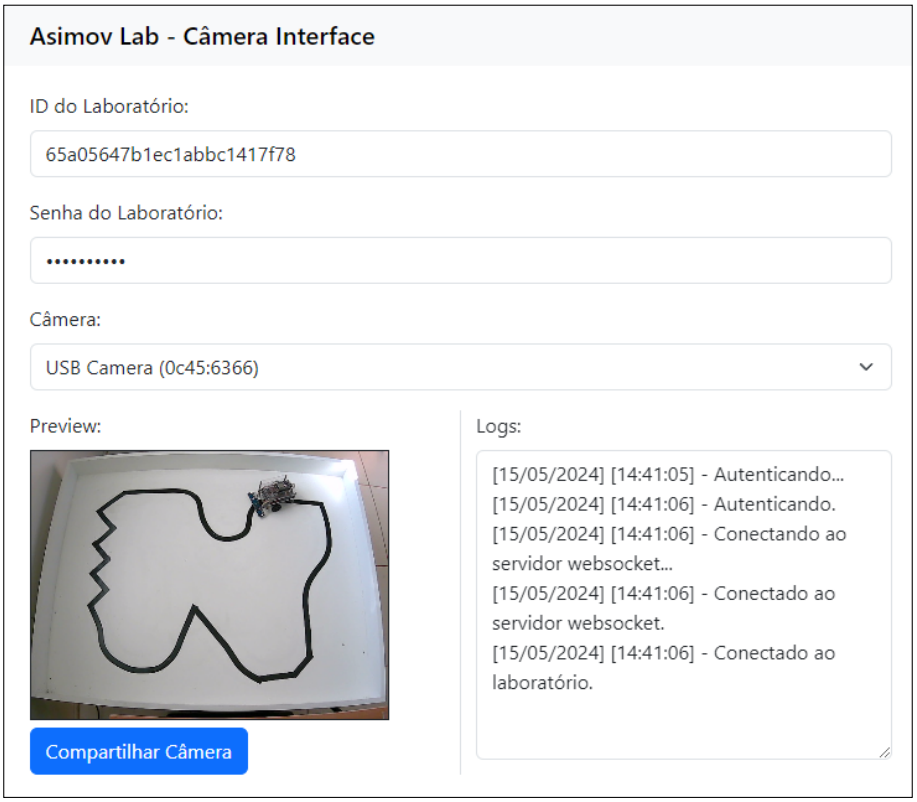

Figura 4. Interface *web* para compartilhamento de *webcam* do experimento local. Esta interface faz parte da aplicação do servidor local. Fonte: Autoria própria.

4.3. Monitor Serial

Monitor serial é uma ferramenta importante para quem realiza experimentos com microcontroladores, pois permite ao usuário enviar e receber dados diretamente na interface de comunicação serial do dispositivo. Isso torna o monitor serial uma ferramenta útil para depuração de programas [Arduino 2024].

No sistema, a transmissão remota de dados entre monitor serial e dispositivo é possível pela conexão física entre os dois pares de portas comunicação RX/TX disponíveis na placa de desenvolvimento ESP32 DEVKIT V1, como ilustrado na Figura 5. Um dos pares (RX2 e TX2) é controlado pela aplicação de controle e o outro par (RX0 e TX0) é disponibilizado para o usuário manipular em seu código. Os dados são transmitidos pela internet através do protocolo *websocket* e uma rotina presente na aplicação de controle na placa de desenvolvimento realiza a retransmissão dos dados para as portas RX/TX da placa. Na Figura 6 é apresentado um esquema geral da comunicação entre o monitor serial e a placa ESP32 DEVKIT V1.

É importante notar que esta abordagem requer o sacrifício de um par de portas RX/TX para o funcionamento adequado do monitor serial. No entanto, a vantagem é a eliminação da necessidade de adaptar as funções de comunicação serial da biblioteca Arduino.

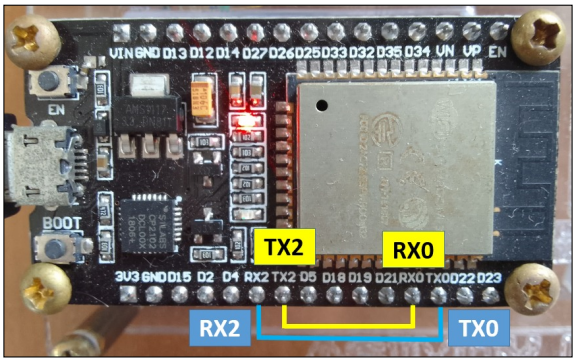

Figura 5. Conexão física entre interfaces de comunicação Serial. Fonte: Autoria própria.

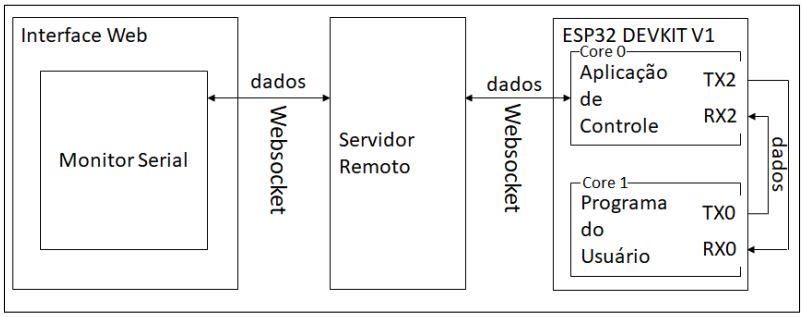

Figura 6. Os dados trafegam pela internet através de *websockets* e são retransmitidos pelos pares de portas RX/TX da placa ESP32 DEVKIT V1 pela aplicação de controle. Fonte: Autoria própria.

4.4. Interface Web

Esta aplicação é responsável por prover a interface gráfica de usuário, e seu acesso é realizado por meio de um navegador web. Em seu desenvolvimento foi utilizado o *framework* Vue.js versão 3.2.38 [Vue 2024] juntamente com a biblioteca de componentes gráficos Vuetify versão 3.4.9 [Vuetify 2024]. Este *framework* foi escolhido por trazer mecanismos importantes para a aplicação, como por exemplo, o sincronismo dos elementos gráficos conforme a mudança de estado das variáveis dos componentes e também o mecanismo que permite o desenvolvimento de aplicação de página única ou SPA (*Single Page Application*). Já a biblioteca Vuetify traz componentes gráficos prontos, baseados no sistema de *design* Material da Google. Optou-se pela utilização de componentes prontos por favorecer a economia de tempo e evitar a preocupação demasiada com o *design* da aplicação. Foi empregado também o editor de código Monaco da Microsoft versão 1.0.10 [Microsoft 2022]. Este editor se destaca por permitir a sua inclusão em uma página *web* como uma biblioteca JavaScript.

4.4.1. Interface de Apresentação e Gerenciamento de Experimentos

Na Figura 7 é exibida a interface gráfica onde são mostrados todos os experimentos disponíveis. Cada experimento possui o seu *hardware* próprio. Nesta interface, estão presentes diversos botões, cujas funcionalidades são descritas abaixo:

A: Inserção de um novo experimento.
B: Apresenta a interface de agendamento do experimento.
C: Apresenta dados de credenciais para serem utilizados pela aplicação do servidor local e dispositivo, para que possam se conectar ao sistema.
D: Acesso a interface de codificação e experimento do laboratório.
E: Alteração de ordem de apresentação dos experimentos.
F: Edição de dados do experimento.
G: Exclusão do experimento.

Os botões **A**, **C**, **E**, **F** e **G** são mostrados apenas para usuários com perfil de administrador do sistema. Já os botões **B** e **D** são mostrados para todos os usuários.

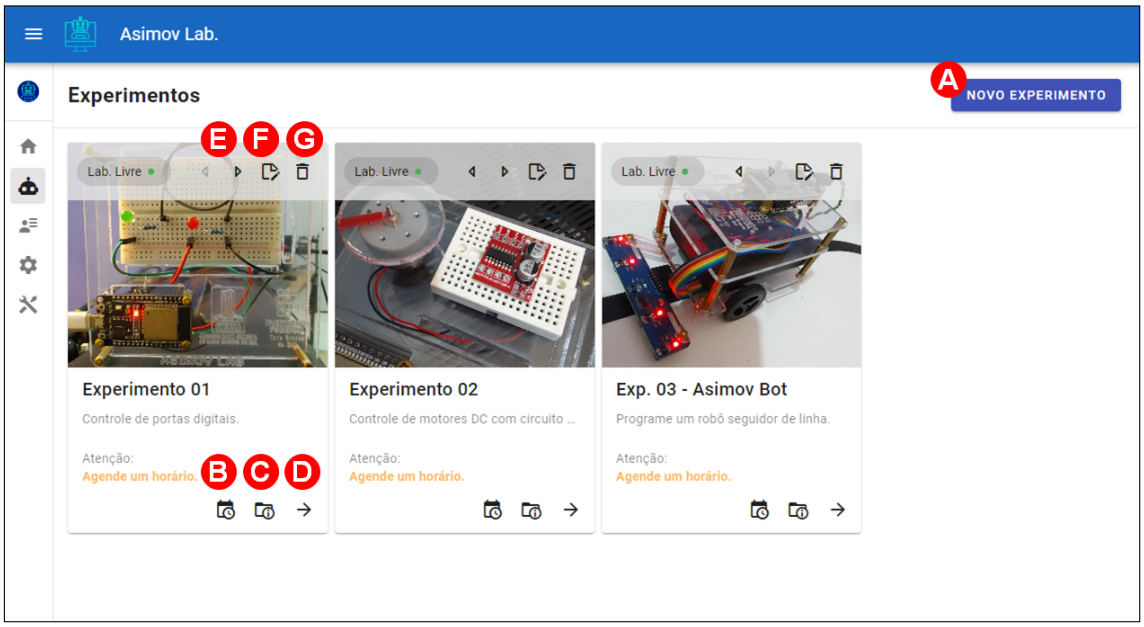

Figura 7. Interface gráfica de gerenciamento de laboratórios. Fonte: Autoria própria.

4.4.2. Interface do Experimento

Esta é a principal interface gráfica do sistema (Fig. 8). Esta interface possibilita ao usuário escrever o seu programa, realizar o *upload* do código para compilação e execução pelo dispositivo, visualizar o experimento em tempo real e interagir com a interface de comunicação serial do dispositivo por meio do monitor serial. Os elementos presentes nessa interface são descritas abaixo:

A: Botão para acesso a interface de gerenciamento de arquivos do usuário.
B: Botão que realiza o *upload* do código do usuário para ser compilado pela aplicação do servidor local. Após a compilação, o programa é baixado pelo dispositivo e executado.
C: Botão que inicia o programa do usuário.
D: Botão que encerra o programa do usuário.
E: Botão que apresenta a janela de feedback visual do experimento (**K**).

F: Botão que apresenta a janela da ferramenta monitor serial (**M**).
G: Informação de estado de conexão do dispositivo com o sistema.
H: Informação de estado de conexão da aplicação do servidor local com o sistema.
I: Informação de tempo restante que o usuário possui para utilizar o experimento. Caso o tempo acabe, o sistema salva automaticamente o código do usuário.
J: Editor de código fonte.
K: Janela responsável por mostrar o feedback visual do experimento.
L: *Joystick* com um controle direcional e dois botões de disparo para interação com dispositivos.
M: Janela da ferramenta monitor serial.
N: Console remoto. Apresenta informações do processo de compilação de código fonte do usuário.

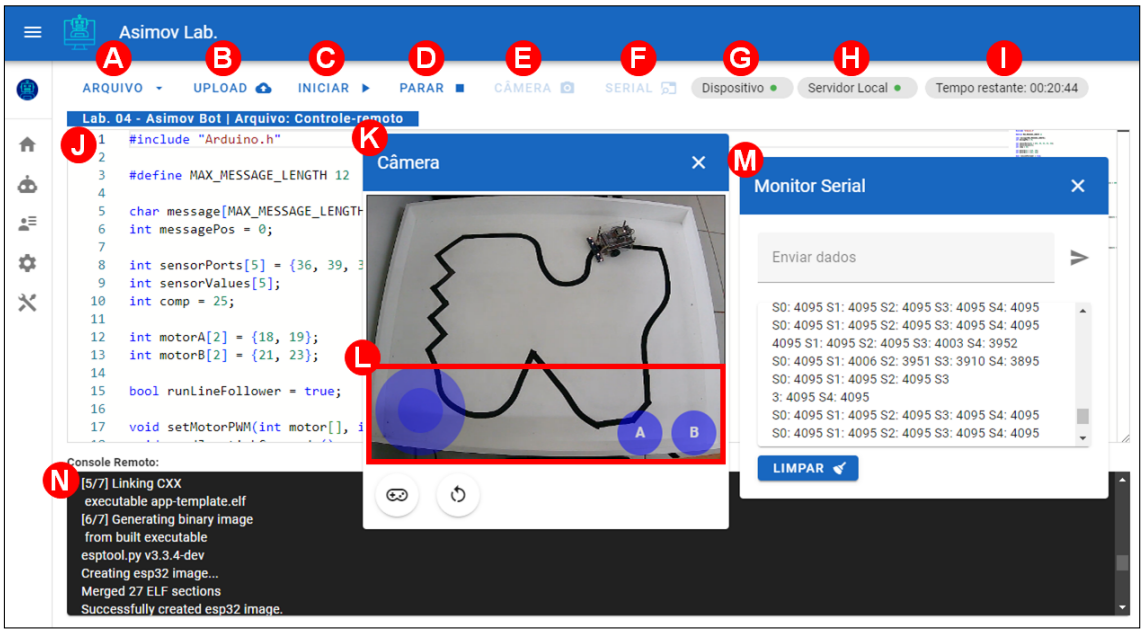

Figura 8. Interface gráfica do experimento. Fonte: Autoria própria.

4.5. Aplicação do Servidor Remoto

A aplicação do servidor remoto tem a finalidade de prover serviços do tipo REST (*Representational State Transfer*), como manutenção de usuários, agendamentos para utilização do laboratório, persistência dos códigos fontes em banco de dados, etc. A aplicação desenvolvida, além de ser um servidor HTTP, também é um servidor *websocket*. O servidor *websocket* permite a comunicação bidirecional entre o servidor e clientes. A comunicação bidirecional é necessária em vários momentos. Destacam-se três: Primeiro, quando o servidor remoto recebe o código fonte da interface web e precisa repassar para o servidor local do laboratório físico para compilação. Segundo, quando os detalhes da compilação precisam ser enviados para a interface web para serem mostrados ao usuário. Terceiro, para troca de dados entre monitor serial e dispositivo.

A aplicação do servidor remoto foi desenvolvida com a plataforma Node.js versão 14.19.1 [Node.js 2022], utilizando a biblioteca express [Express.js 2022] para facilitar a implementação de serviços HTTP tipo REST e a biblioteca ws [WebSockets 2022] que

permite levantar um servidor *websocket* e tratar seus eventos. Para a persistência de dados foi utilizado o banco de dados orientado a documentos MongoDB.

4.6. Aplicação do Servidor Local

O servidor local é executado em um computador localizado no laboratório físico, e sua principal função é compilar o código fonte do usuário e disponibilizar o novo *software* para ser baixado pelo microcontrolador ESP32 por meio de requisição HTTP. Esse servidor é necessário pois a compilação do código fonte necessita da instalação de ferramentas especificas do microcontrolador ESP32.

A aplicação do servidor local foi desenvolvida na plataforma Node.js, utilizando a biblioteca express e a biblioteca ws. Para o seu funcionamento, foi necessário a instalação de uma ferramenta denominada de *toolchain* para compilação e construção dos softwares para microcontroladores da família ESP32. Além do *toolchain*, também foi necessário instalar a biblioteca ESP-IDF versão 4.4.1.

O seu funcionamento ocorre da seguinte forma: a aplicação do servidor local recebe o código fonte do servidor remoto por meio de *websocket* e o salva em um arquivo com extensão c, logo após, a aplicação executa um arquivo em lote que possuí comandos de compilação específicos das ferramentas do ESP32. Após a compilação, a aplicação do servidor local envia uma mensagem por meio de *websocket* ao microcontrolador informando que um novo *software* está disponível para ser baixado.

4.7. Aplicação de Controle

Os microcontroladores da família ESP32 permitem que um novo *software* seja carregado por meio de conectividade sem fio, sem a necessidade de conexão por meio de cabos. Esse mecanismo é chamado de atualização pelo ar, ou *Update Over-the-Air* (OTA). Para que o mecanismo de atualização OTA funcione de forma correta, é exigida a criação de duas partições a mais na memória de armazenamento do ESP32. As duas partições são necessárias pois funcionam em alternância, enquanto uma das partições armazena o *software* em execução, a outra armazena o novo *software* baixado. Além disso, é necessário que o *software* inicial e posteriores estejam preparados para verificar a existência de uma nova versão do *software* em um servidor para ser baixado e armazenado em uma das partições.

Como o ESP32 possui dois núcleos de processamento, a aplicação de controle foi dividida em duas tarefas, cada tarefa é executada em núcleos distintos. A primeira tarefa, que é executada no primeiro núcleo, está relacionada com as atividades de conexão com a rede, conexão com o servidor remoto e atualização de *software*, o fluxo de execução é exemplificado no fluxograma apresentado na Figura 9. A segunda tarefa, que é executada no segundo núcleo, tem como objetivo executar o programa do usuário. O programa de usuário é decorrente do código fonte submetido pelo usuário por meio da interface *web* e segue o padrão utilizado pela plataforma Arduino, que estabelece duas funções fundamentais: `setup()` e `loop()`. A aplicação de controle executa, primeiramente, a função `setup()` e, logo após, executa a função `loop()` repetidas vezes. A execução é exemplificada no fluxograma apresentado na Figura 10.

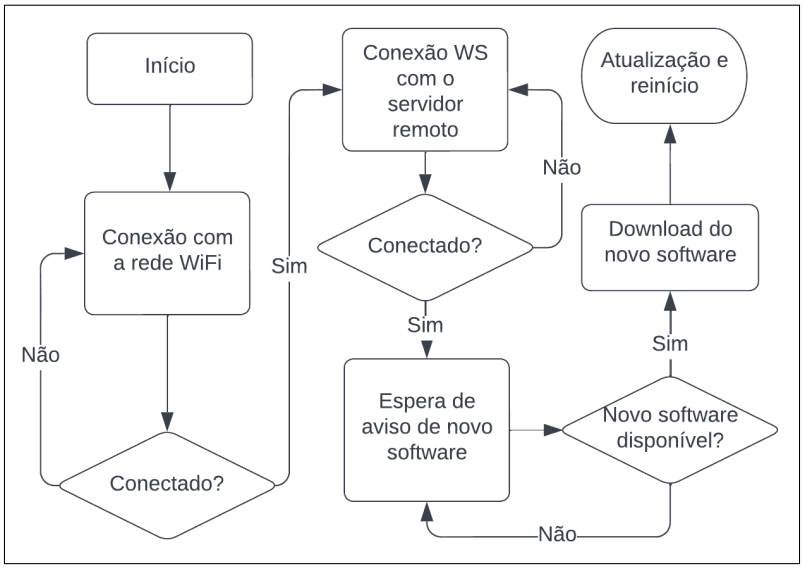

Figura 9. Fluxograma da tarefa principal da aplicação de controle. Fluxo executado pelo primeiro núcleo do processador do ESP32. Fonte: Autoria própria.

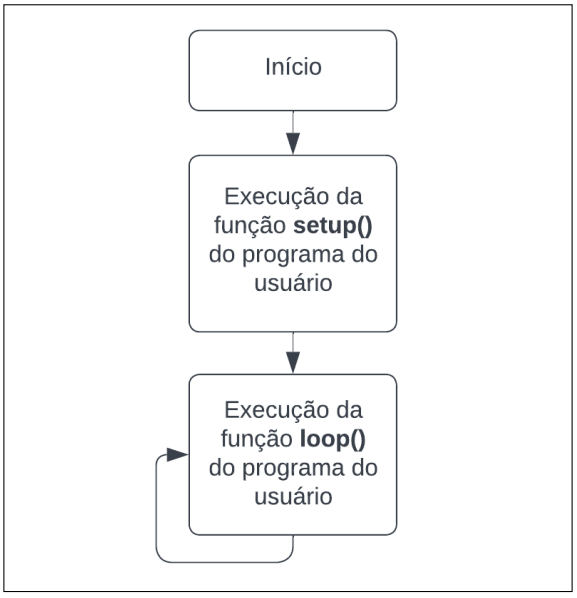

Figura 10. Fluxograma da tarefa secundária da aplicação de controle. Fluxo executado pelo segundo núcleo do processador do ESP32. Fonte: Autoria própria.

5. Teste

Com intuito de testar o sistema em ambiente de produção, foi elaborado um curso de programação e robótica para alunos do curso técnico em desenvolvimento de sistemas, alunos do curso superior em sistemas para internet e docentes da área de computação de ambos os cursos do Instituto Federal de Educação, Ciência Tecnologia de Mato Grosso do Sul (IFMS), campus Coxim. O curso foi ofertado totalmente a distância, com aulas gravadas[1] e disponibilizadas no ambiente virtual de aprendizagem da instituição, o sistema Moodle. O sistema desenvolvido neste trabalho, permitiu que a parte prática do

[1]Vídeos disponíveis: `https://bit.ly/4cPP6Wg`

curso fosse realizada a distância pelos participantes. Este curso foi fundamental para que os participantes interagissem com o sistema e adquirissem embasamento para avalia-lo.

5.1. Conteúdo Programático do Curso

No curso foram abordados conceitos básicos de programação e técnicas para que, no final, os participantes tenham habilidades suficientes para programar um robô seguidor de linha. Os seguintes assuntos fazem parte do conteúdo programático do curso:

- Microcontroladores e robôs.
- ESP-32 e plataforma Arduino.
- Tour pelo sistema Asimov Lab.
- Funções básicas: setup, loop, pinMode, digitalWrite e delay.
- Comunicação serial.
- Variáveis.
- Operadores aritméticos, relacionais e lógicos.
- Estruturas condicionais.
- Estruturas de repetição.
- Arrays.
- Strings.
- Controle de motores DC com circuito ponte H.
- Leitura de sensores analógicos.
- Programação de um robô seguidor de linha.

5.2. Experimentos

Foram desenvolvidos três experimentos especialmente para o curso, cada experimento possui o seu dispositivo físico. O primeiro experimento (Figura 11-**A**) foi elaborado para abordar os conceitos introdutórios de programação e funções básicas da biblioteca Arduino. O segundo experimento (Figura 11-**B**) foi elaborado para a prática de controle de motores DC com circuito ponte H. O terceiro experimento se trata de um robô seguidor de linha (Figura 12-**A**) acompanhado de uma pista (Figura 12-**B**), no qual o cursista tem a oportunidade de colocar em prática o conhecimento adquirido durante o curso.

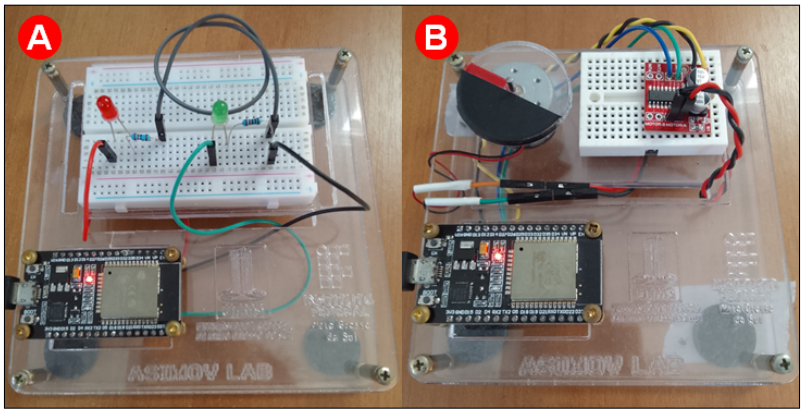

Figura 11. Dispositivos dos experimentos 1 e 2. A - circuito simples com dois leds. B - circuito com um motor DC e modulo de ponte H dupla L298N. Fonte: Autoria própria.

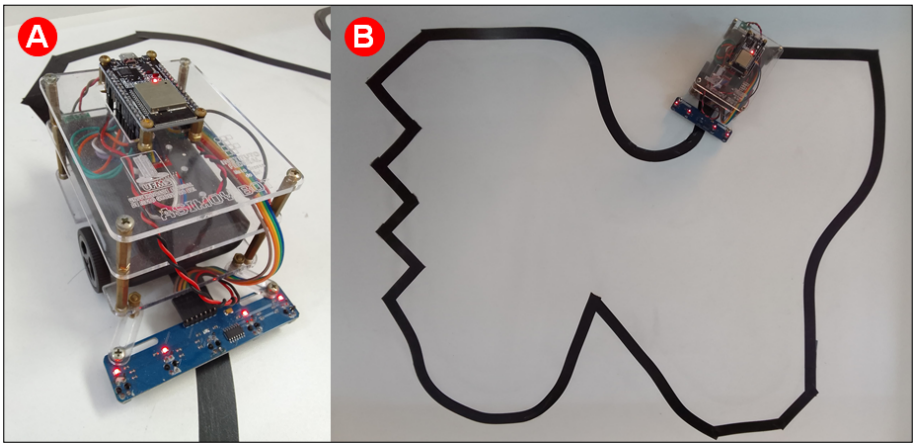

Figura 12. Dispositivo do experimento 3. A - Robô seguidor de linha com *array* de sensores reflexivos infravermelho, par de motores N20 75 RPM 3v, ponte H dupla L298N e *powerbank* de 5v e 10000 mAH. B - Pista para o robô seguidor de linha. Fonte: Autoria própria.

6. Discussão dos Resultados

Durante o andamento do curso, foi disponibilizado um questionário por meio da ferramenta Google Forms, para a avaliação do sistema Asimov - Laboratório Remoto de Robótica (Tabela 1). O questionário foi baseado nos trabalhos de Valentim et al. (2014) e Finger et al. (2021) . Ambos os trabalhos tem como finalidade o teste de usabilidade de *softwares* relacionados a educação e baseiam-se no paradigma GQM (*Goal-Requestion-Metric*) [Caldiera and Rombach 1994], levando em consideração a percepção do usuário em relação a utilidade, facilidade de uso, interface gráfica e ensino em sala de aula. Importante salientar que o final da questão **17** (Tabela 1) foi alterado, de "...atividades em sala." conforme o questionário original presente nos trabalhos de Valentim et al. (2014) e Finger et al. (2021) , para "...atividades em EaD.", pois todo o curso foi aplicado em modalidade EaD (Educação à Distância). Foram incluídas também, duas questões abertas, uma para relatos de problemas e sugestões.

A escala utilizada para as respostas do questionário foi baseada na escala de Likert (1932). A escala compreende em cinco tipos de respostas, que correspondem a intensidade de concordância com as questões. As respostas possíveis aplicadas no questionário consistem em: 1 - Discordo totalmente, 2 - Discordo parcialmente, 3 - Indiferente, 4 - Concordo parcialmente e 5 - Concordo totalmente. Esta escala permitiu medir a percepção dos usuários sobre os escopos (Facilidade de Uso, Utilidade, Interface Gráfica e Ensino) conforme a experiência de uso do sistema.

Ao todo foram 66 candidatos que manifestaram interesse em realizar o curso. Sendo 22 estudantes do curso Superior em Tecnologia em Sistemas para Internet, 34 do curso Técnico em Desenvolvimento de Sistemas e 10 docentes de áreas relacionadas a computação. Dos 66 cursistas, responderam o questionário: 11 estudantes do curso superior, 13 estudantes do curso técnico e 9 docentes. Ressalta-se que a realização do curso e o preenchimento do questionário não foram obrigatórios. As discussões sobre os resultados foram divididas em subseções conforme os escopos das questões: Facilidade de uso (subseção 6.1), Utilidade (subseção 6.2), Interface Gráfica (subseção 6.3), Ensino (subseção 6.4), Problemas (subseção 6.5) e Sugestões (subseção 6.6).

Tabela 1. Questionário aplicado para a avaliação do sistema.

Escopo	Nº	Questão
Facilidade de Uso	Q1	Foi fácil aprender a utilizar o software.
	Q2	Eu conseguia entender o que acontecia durante o uso do software.
	Q3	Foi fácil ganhar habilidade de uso durante a execução das atividades no software.
	Q4	É fácil de lembrar como utilizar o software.
	Q5	Considero o software fácil de utilizar.
Utilidade	Q6	Considero o software útil para melhorar meu aprendizado.
	Q7	Considero que o software melhoraria minha produtividade para realização das atividades e aprendizado.
Interface Gráfica	Q8	Considero as cores e botões do software agradáveis.
	Q9	Consigo visualizar bem todos os botões e informações dentro do software.
	Q10	Entendo com facilidade as palavras, nomenclaturas e ícones do software.
	Q11	As imagens e ícones no software são de fácil reconhecimento.
	Q12	Consigo visualizar todas as funcionalidades do software.
	Q13	Consigo navegar bem por todas as telas do software.
Ensino	Q14	Após me acostumar com o software julgo que o software facilita o ensino e aprendizagem
	Q15	Com o software considero que as aulas ficariam mais interessantes.
	Q16	Com o software eu conseguiria explorar bem os materiais disponíveis
	Q17	Com o software eu teria maior domínio sobre as atividades em EaD
	Q18	Com o software o nível de aprendizagem aumentaria

6.1. Facilidade de Uso

As questões referentes a facilidade de uso permitem verificar a percepção dos usuários sobre o nível dificuldade que tiveram para aprender e lembrar como manipular o sistema [Finger et al. 2021].

O gráfico apresentado na Figura 13, mostra que em média 83% dos estudantes (20 de 24) concordam parcialmente ou totalmente com questões relacionadas a facilidade de uso. Já o gráfico da Figura 14 mostra que em média 100% dos docentes (9) concordam parcialmente ou totalmente com questões relacionadas a facilidade de uso. A hipótese que pode justificar esta diferença, é que os docentes possuem um conhecimento tácito de utilização de sistemas por serem de áreas relacionadas a computação. Notou-se também que a vídeo aula do curso, na qual é ensinado a como operar o sistema, contribuiu para a boa avaliação no quesito facilidade de uso, conforme relato de um dos estudante nas questões abertas.

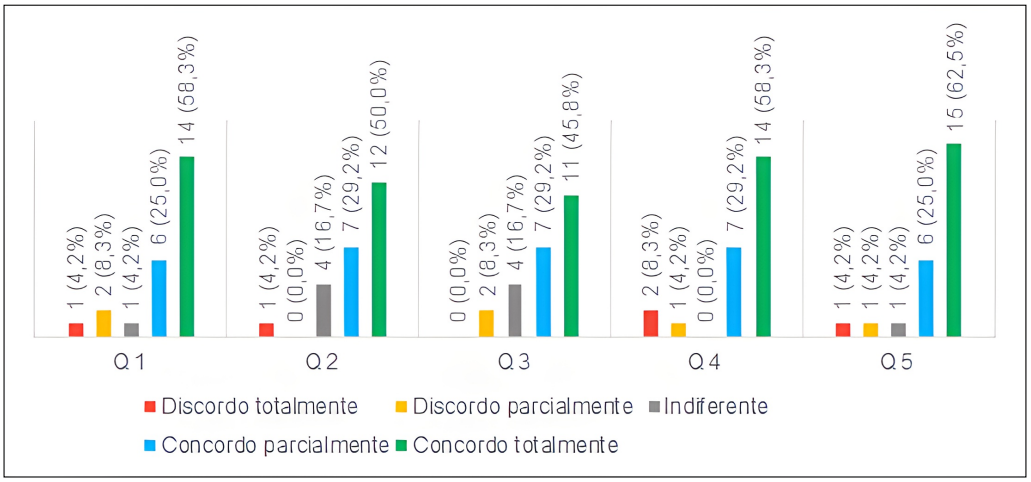

Figura 13. Escopo Facilidade de Uso - Estudantes. Fonte: Autoria própria.

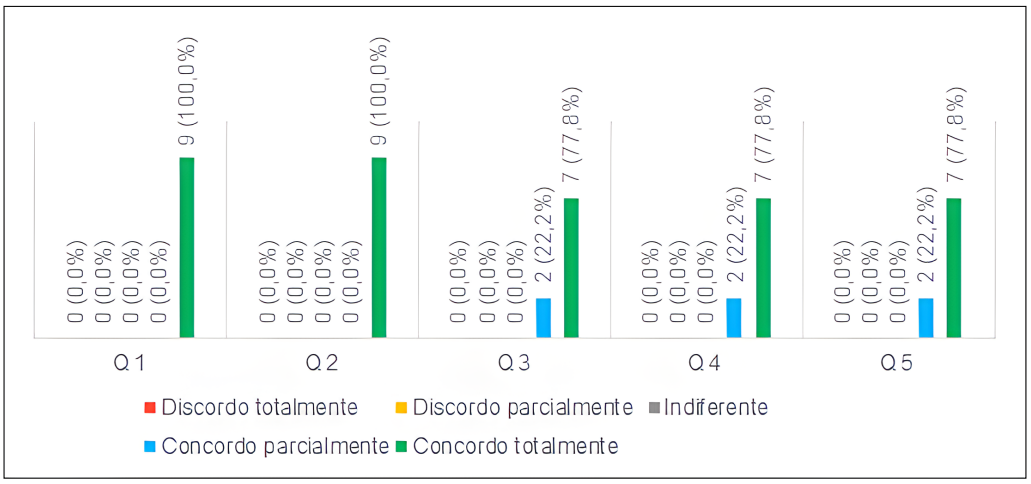

Figura 14. Escopo Facilidade de Uso - Docentes. Fonte: Autoria própria.

6.2. Utilidade

As questões deste escopo permitem verificar se os usuários consideram o sistema útil como uma ferramenta de estudos para melhoria de aprendizado e produtividade para realização de atividades [Finger et al. 2021].

O gráfico apresentado na Figura 15, mostra que em média 90% dos estudantes (22 de 24) concordam parcialmente ou totalmente que o sistema é útil como uma ferramenta de estudos. O gráfico da Figura 16 mostra a percepção dos docentes, sendo que em média, todos concordam parcialmente ou totalmente que o sistema é uma ferramenta útil para estudos. Tais números indicam que o sistema tem potencial para ser uma alternativa de ferramenta para agregar nos estudos de robótica e programação para microcontroladores, tanto para alunos, quanto para docentes.

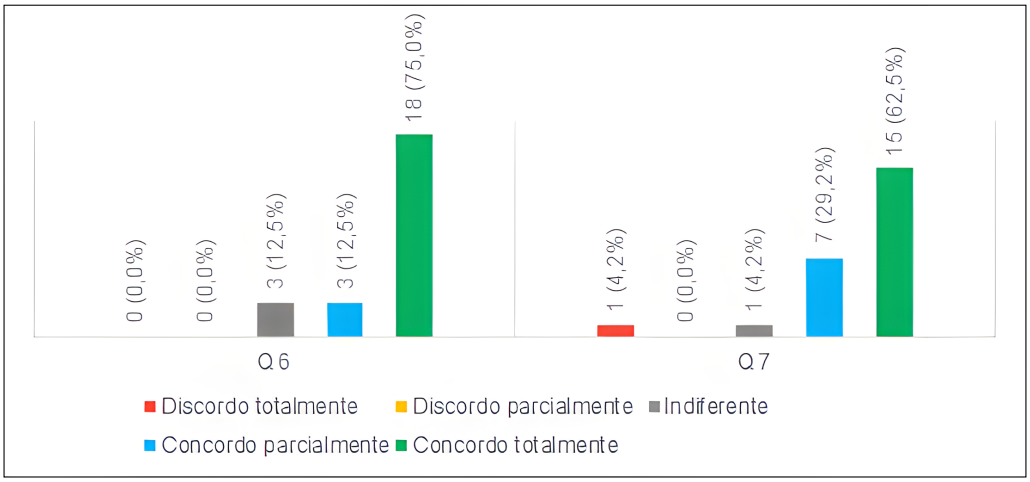

Figura 15. Escopo Utilidade - Estudantes. Fonte: Autoria própria.

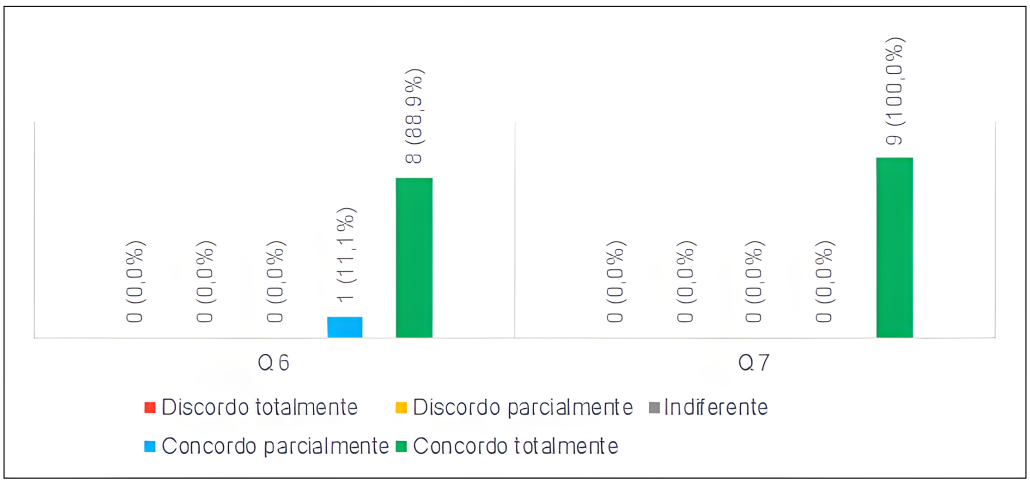

Figura 16. Escopo Utilidade - Docentes. Fonte: Autoria própria.

6.3. Interface Gráfica

As questões relacionadas a *interface* gráfica levam em consideração a percepção dos usuários em relação a navegabilidade e elementos visuais do sistema como: botões, ícones, cores, imagens e informações [Finger et al. 2021].

Observa-se no gráfico da Figura 17, que 76% dos estudantes (18 de 24) concordam parcialmente ou totalmente com a adequação dos elementos que compõem a interface gráfica do sistema. O gráfico da Figura 18 mostra a percepção dos docentes, sendo que todos concordam parcialmente ou totalmente com a adequação da interface gráfica.

A interface gráfica é um ponto que merece mais atenção para atualizações, pois dos três escopos (facilidade de uso, utilidade e interface gráfica) avaliados pelos estudantes, este foi o que apresentou o número mais baixo de concordância.

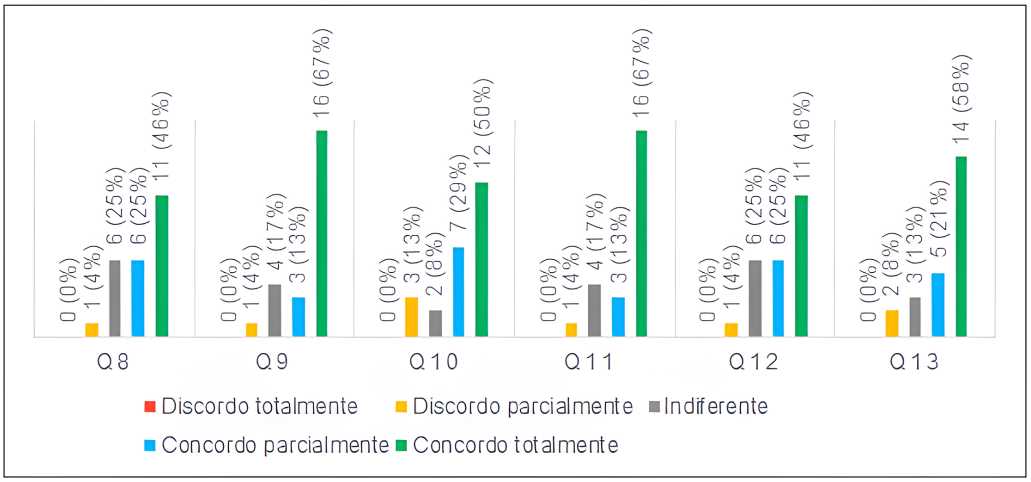

Figura 17. Escopo Interface Gráfica - Estudantes. Fonte: Autoria própria.

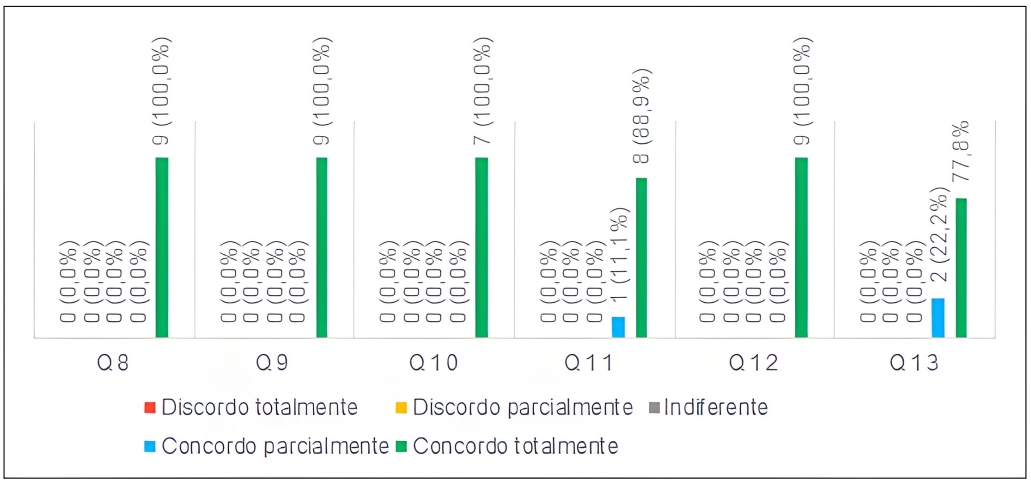

Figura 18. Escopo Interface Gráfica - Docentes. Fonte: Autoria própria.

6.4. Ensino

As questões relacionadas ao ensino levam em consideração a possibilidade de utilização do sistema em ambiente de ensino como uma ferramenta para tornar as aulas mais interessantes, facilitar o ensino e aprendizagem e utilizar em atividades em educação à distância. Estas questões foram aplicadas apenas aos docentes.

O gráfico da Figura 19 mostra que 100% dos docentes (9) concordam parcialmente ou totalmente que o sistema pode ser utilizado em âmbito educacional. Neste contexto, o sistema pode ser útil em disciplinas relacionadas a robótica, microcontroladores e programação, bem como em projetos de pesquisa ou extensão. Um dos relatos nas questões abertas afirma que o sistema será muito útil nas atividades de ensino, principalmente quando não se têm acesso aos componentes físicos nas escolas, além de possibilitar desenvolver atividades e observar o comportamento dos componentes na prática.

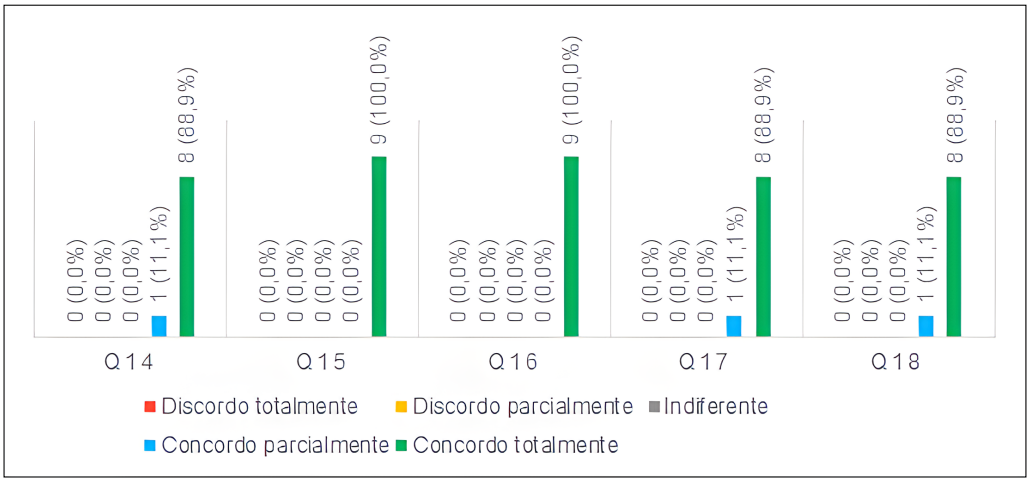

Figura 19. Escopo Ensino - Docentes. Fonte: Autoria própria.

6.5. Problemas

Alguns problemas foram relatados nas questões abertas. Os relatos de problemas foram sintetizados para melhor apresentação e podem ser conferidos na Tabela 2. Tais relatos ajudarão na correção para versões futuras do sistema, pois há interesse em compartilhar os laboratórios e também aplicar o curso em outras instituições de ensino, então é importante garantir a estabilidade do sistema.

Tabela 2. Síntese dos problemas relatados pelos participantes.

Nº	Problema
P1	Confusão com o ícone do botão de upload de código. O ícone deu a entender que o código seria apenas salvo em nuvem e não ser compilado e executado.
P2	Joystick atrapalha a visualização da câmera.
P3	Utilização do tradutor do navegador está traduzindo o código fonte.
P4	Visualização da câmera do laboratório 02 sumia.
P5	Demora na compilação do código fonte.

6.6. Sugestões

Importantes sugestões foram dadas nas questões abertas. As sugestões foram sintetizadas e são apresentadas na Tabela 3. Destaca-se a sugestão **S5** que trata da inclusão de compatibilidade com a plataforma MicroPython. Esta plataforma é uma implementação otimizada da linguagem de programação Python 3 e suas bibliotecas para microcontroladores. MicroPhyton permite a compilação do código em tempo de execução, tornando a compilação e execução do programa do usuário mais rápida [MicroPython 2024]. Isso mitigaria o problema **P5** da Tabela 2, pois com as ferramentas nativas do ESP32, o *firmware* é compilado por completo e gravado na memória flash, ocasionando a demora relatada.

Tabela 3. Síntese das sugestões dos participantes.

Nº	Sugestão
S1	Incluir um ícone para baixar o código fonte.
S2	Disponibilizar mais experimentos.
S3	Módulo do sistema onde o professor/instrutor possa cadastrar atividades e monitorar a realização destas pelos estudantes.
S4	Exercícios e correção automática, integração com as notas no sistema moodle
S5	Incluir compatibilidade com a plataforma MicroPython.
S6	Incluir a funcionalidade de redimensionamento da janela da câmera.
S7	Incluir manual do usuário.
S8	Incluir gerenciamento de arquivos na página principal, atualmente está presente apenas quando entra na interface de algum experimento.
S9	Incluir ajuste de tempo de utilização do experimento. Atualmente está fixo em 30 minutos para usuário sem agendamento e 60 minutos para usuários com agendamento
S10	Incluir informação de versão do código que está em execução no momento em que o usuário está testando o experimento.

7. Conclusão

Laboratórios remotos podem ser úteis em situações que inviabilizem o acesso às dependências físicas das instituições de ensino, como foi o caso ocorrido durante a pandemia de COVID-19, que levou muitas instituições de ensino a suspenderem o acesso presencial, tornando os laboratórios ociosos. Além disso, os laboratórios remotos permitem que os recursos presentes em um laboratório físico sejam compartilhados com outras instituições que não possuem laboratórios específicos.

Durante o desenvolvimento do sistema Asimov - Laboratório Remoto de Robótica, foram exploradas algumas tecnologias que podem ser uteis para o desenvolvimento de outros sistemas. Como por exemplo o protocolo WebRTC que já vem implementado em diversos navegadores de internet e permite conexão ponto a ponto entre navegadores de internet, com a finalidade de trafegar dados de vídeo e áudio com menos latência possível.

O sistema obteve resultados satisfatórios nas avaliações realizadas por estudantes e docentes que participaram o curso básico de programação e robótica ofertado. Tais resultados evidenciam que o sistema pode se tornar uma ferramenta útil para fins educacionais. Podendo ser utilizado para cursos e disciplinas relacionadas a programação de microcontroladores e robótica, principalmente em contexto EaD. Os resultados obtidos irão ajudar a dar continuidade no desenvolvimento do sistema, na correção de erros e implementação de novas funcionalidades. Importante ressaltar que este sistema não substitui sistemas de laboratórios virtuais, mas sim, servir como uma opção de ferramenta para agregar o ambiente educacional.

Como trabalhos futuros, indica-se a implementação de compatibilidade com a plataforma MicroPython como sugerido na subseção 6.6 da discussão dos resultados. Outra compatibilidade interessante para ser implementada, seria com a plataforma Lego Mindstorm, pois esta plataforma é muito utilizada em contexto educacional, porém possui um

custo elevado para aquisição.

Agradecimentos

O presente trabalho foi realizado com apoio da Universidade Federal de Mato Grosso do Sul - UFMS/MEC - Brasil, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001, Faculdade de Computação FACOM/UFMS, Instituto Federal de Educação Ciência e Tecnologia de Mato Grosso do Sul - IFMS e IFMaker - IFMS Campus Coxim.

Referências

Almeida, T. O., de M. Netto, J. F., and Rios, M. L. (2017). Remote robotics laboratory as support to teaching programming. In *2017 IEEE Frontiers in Education Conference (FIE)*, pages 1–6.

Alvestrand, H. T. (2021). Overview: Real-Time Protocols for Browser-Based Applications. RFC 8825.

Andriola, W. B. (2021). Impactos da robótica no ensino básico: estudo comparativo entre escolas públicas e privadas. *Ciência & Educação (Bauru)*, 27.

Arduino (2024). Using the arduino ide 2.0 serial monitor. `https://docs.arduino.cc/software/ide-v2/tutorials/ide-v2-serial-monitor/`. Acessado em 16 de maio de 2024.

Bavel, J. J. V., Baicker, K., Boggio, P. S., Capraro, V., Cichocka, A., Cikara, M., Crockett, M. J., Crum, A. J., Douglas, K. M., Druckman, J. N., Drury, J., Dube, O., Ellemers, N., Finkel, E. J., Fowler, J. H., Gelfand, M., Han, S., Haslam, S. A., Jetten, J., Kitayama, S., Mobbs, D., Napper, L. E., Packer, D. J., Pennycook, G., Peters, E., Petty, R. E., Rand, D. G., Reicher, S. D., Schnall, S., Shariff, A., Skitka, L. J., Smith, S. S., Sunstein, C. R., Tabri, N., Tucker, J. A., Linden, S. v. d., Lange, P. v., Weeden, K. A., Wohl, M. J. A., Zaki, J., Zion, S. R., and Willer, R. (2020). Using social and behavioural science to support covid-19 pandemic response. *Nature Human Behaviour*, 4(5):460–471.

Caldiera, V. R. B. G. and Rombach, H. D. (1994). The goal question metric approach. *Encyclopedia of software engineering*, pages 528–532.

Campos, F. R. (2017). Robótica educacional no brasil: questões em aberto, desafios e perspectivas futuras. *Revista ibero-americana de estudos em educação*, 12(4):2108–2121.

Coturn (2024). Coturn turn server. `https://github.com/coturn/coturn`. Acessado em 15 de maio de 2024.

Dargains, A. R. and Sampaio, F. F. (2020). Estudo exploratório sobre o uso da robótica educacional no ensino de introdução a programação. *Tecnologias, Sociedade e Conhecimento*, 7(1):71–96.

Di Giamberardino, P. and Temperini, M. (2017). Adaptive access to robotic learning experiences in a remote laboratory setting. In *2017 18th International Carpathian Control Conference (ICCC)*, pages 565–570.

Espressif (2022a). Esp-idf programming guide. https://docs.espressif.com/projects/esp-idf/en/v4.4.1/esp32/esp-idf-en-v4.4.1-esp32.pdf. Acessado em 23 de abril de 2022.

Espressif (2022b). Esp32-wroom-32: Datasheet. https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf. Acessado em 23 de abril de 2022.

Express.js (2022). 4.x api. https://expressjs.com/pt-br/4x/api.html. Acessado em 23 de abril de 2022.

Finger, A., Loreto, A., Soubhia, A. L., and Andrade, A. (2021). Avaliação de usabilidade do sofemn: Software de apoio ao ensino de métodos numéricos. In *Anais do XXXII Simpósio Brasileiro de Informática na Educação*, pages 103–112, Porto Alegre, RS, Brasil. SBC.

Flores, C., Kronbauer, A., and Campos, J. (2018). Lero-an extensible and adaptive remote lab for educational robotics. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 29, page 525.

Gomes, L. and Bogosyan, S. (2009). Current trends in remote laboratories. *IEEE Transactions on Industrial Electronics*, 56(12):4744–4756.

Kerem Erdoğmuş, A. and Yayan, U. (2021). Virtual robotic laboratory compatible mobile robots for education and research. In *2021 International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*, pages 1–6.

Likert, R. (1932). A technique for the measurement of attitudes. *Archives of psychology*.

Lin, M., San, L., and Ding, Y. (2020). Construction of robotic virtual laboratory system based on unity3d. *IOP Conference Series: Materials Science and Engineering*, 768(7):072084.

Maruyama, G. Y., de Castro Junior, A. A., de Lima, A. C., and Vilhanueva, M. P. (2022). Tecnologias para implementação de laboratórios remotos de robótica: Revisão sistemática da literatura. In *Anais do Computer on the Beach*, pages 59–65, Itajaí, SC, Brasil.

Matthews, P., Rosenberg, J., and Mahy, R. (2010). Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN). RFC 5766.

MicroPython (2024). Micropython documentation. https://docs.micropython.org/en/latest/reference/index.html. Acessado em 21 de maio de 2024.

Microsoft (2022). About. https://microsoft.github.io/monaco-editor/. Acessado em 23 de abril de 2022.

Nedic, Z., Machotka, J., and Nafalski, A. (2003). Remote laboratories versus virtual and real laboratories. In *33rd Annual Frontiers in Education, 2003. FIE 2003.*, volume 1, pages T3E–T3E.

Node.js (2022). Node.js v14.19.1 documentation. https://nodejs.org/docs/latest-v14.x/api/. Acessado em 23 de abril de 2022.

Orduna, P., Rodriguez-Gil, L., Garcia-Zubia, J., Angulo, I., Hernandez, U., and Azcuenaga, E. (2016). Labsland: A sharing economy platform to promote educational re-

mote laboratories maintainability, sustainability and adoption. In *2016 IEEE Frontiers in Education Conference (FIE)*, pages 1–6. IEEE.

Souza, C., Paiva, J., Leal, T., Silva, L., and Santos, L. (2020). Spatiotemporal evolution of case fatality rates of covid-19 in brazil, 2020. *J Bras Pneumol.*, 46:e20200208–e20200208.

Valentim, N., Rabelo, J., Silva, W., Coutinho, W., Álvaro Mota, and Conte, T. (2014). Avaliando a qualidade de um aplicativo web móvel através de um teste de usabilidade: um relato de experiência. In *Anais do XIII Simpósio Brasileiro de Qualidade de Software*, pages 256–263, Porto Alegre, RS, Brasil. SBC.

Vidal, J., Sampaio, F., and Dorotea, N. (2021). Um estudo exploratório sobre o uso da robótica educacional como ferramenta de apoio ao ensino-aprendizagem de lógica de programação para alunos da rede pública do ensino médio. In *Anais do Simpósio Brasileiro de Educação em Computação*, pages 280–289, Porto Alegre, RS, Brasil. SBC.

Vue (2024). Vue.js - what is vue. `https://vuejs.org/guide/introduction.html#what-is-vue`. Acessado em 17 de maio de 2024.

Vuetify (2024). Vuetify - why vuetify? `https://vuetifyjs.com/en/introduction/why-vuetify/#why-vuetify`. Acessado em 17 de maio de 2024.

WebSockets (2022). A node.js websocket library. `https://github.com/websockets/ws`. Acessado em 23 de abril de 2022.