CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA

ESCOLA TÉCNICA ESTADUAL DA ZONA LESTE

MTec Desenvolvimento De Sistemas AMS

Guilherme Vinicius de Oliveira

Cássio Egídio Gomes Vicente

Gabriel Anjos de Almeida

TLens: Sistema IoT Para Auxiliar Deficientes Auditivos a Se Comunicarem com Ouvintes

Guilherme Vinicius de Oliveira

Cássio Egídio Gomes Vicente

Gabriel Anjos de Almeida

TLens: Sistema IoT para Auxílio à Comunicação entre Deficientes Auditivos e Ouvintes.

Trabalho de Conclusão de Curso apresentado ao MTec Desenvolvimento de Sistemas AMS da Etec Da Zona Leste, orientado pelo Prof. Esp. Jeferson Roberto de Lima, como requisito parcial para a obtenção do título de técnico em Desenvolvimento de Sistemas

TLens

Sistema IoT para Auxílio à Comunicação entre Deficientes Auditivos e Ouvintes.

Guilherme Vinicius de Oliveira

Cássio Egídio Gomes Vicente

Gabriel Anjos de Almeida

Aprovada em \_\_/\_\_/\_\_\_\_.

BANCA EXAMINADORA

Prof. Esp. Jeferson Roberto de Lima

Universidade do Jeferson

Prof. (Professor avaliador)

Universidade do Avaliador

Prof. (Professor avaliador)

Universidade do Avaliador

AGRADECIMENTOS

Agradeço a xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

RESUMO

O TLens é um sistema vestível e baseado em IoT, com o intuito de mediar a comunicação entre surdos e ouvintes de um modo prático, com custo-benefício e que seja acessível. O aparelho, acoplado a um par de óculos, captura a fala ambiente, e por meio de um modelo de reconhecimento de voz baseado em IA, ele transcreve o texto, projetando-o nas lentes usando tecnologia óptica do tipo *Bird Bath*. A metodologia adotada baseou-se em uma pesquisa qualitativa sobre os problemas mais frequentes que as pessoas com deficiência auditiva encontram linkados à interação verbal, a fim de embasar o projeto. Ainda com os mesmos dados, pode-se identificar os requisitos do sistema, selecionar componentes de baixo custo e fazer a prototipação do produto com foco em usabilidade e acessibilidade. A problemática que motivou este trabalho está relacionada às barreiras da comunicação que dificultam a inclusão social de pessoas com deficiência auditiva. A expectativa é de que a ideia venha a contribuir para a autonomia comunicacional dos usuários, facilitar a inclusão social, remover obstáculos diários enfrentados por essa população e que ela promova o desenvolvimento de novas soluções tecnológicas assistivas.

**Palavras-chave**: inclusão social; deficiência auditiva; projeção óptica; inteligência artificial; IoT.

ABSTRACT

The TLens is a wearable and IoT-based system designed to mediate communication between deaf and hearing individuals in a practical, cost-effective, and accessible way. The device, attached to a pair of glasses, captures ambient speech and, through an AI-based voice recognition model, transcribes the text, projecting it onto the lenses using Bird Bath optical technology. The methodology adopted was based on qualitative research about the most frequent problems that people with hearing impairments face in verbal interaction, in order to support the project. Using the same data, it was possible to identify the system requirements, select low-cost components, and prototype the product with a focus on usability and accessibility. The issue that motivated this work is related to the communication barriers that hinder the social inclusion of people with hearing impairments. The expectation is that the idea will contribute to the communicational autonomy of users, facilitate social inclusion, remove daily obstacles faced by this population, and promote the development of new assistive technological solutions.

**Keywords:** social inclusion; hearing impairment; optical projection; artificial intelligence; IoT.

LISTA DE ILUSTRAÇÕES

[Figura 1 - Modelo 3D Haste de Óculos 18](#_Toc212060965)

[Figura 2 - Orange Pi Zero 2W 19](#_Toc212060966)

[Figura 3 - Módulo de Exibição OLED I2C 20](#_Toc212060967)

[Figura 4 - Lentes Semirrefletivas Divisoras de Feixe 21](#_Toc212060968)

[Figura 5 - Bateria de Li-ion 22](#_Toc212060969)

[Figura 6 - Módulo Controlador de Carga 23](#_Toc212060970)

[Figura 7 - Módulo Regulador de Tensão Step-Up 23](#_Toc212060971)

[Figura 8 - Módulo de Microfone AGC Amplificador MAX9814 24](#_Toc212060972)

[Figura 9 - Exemplo Shell Script 26](#_Toc212060973)

[Figura 10 - Código Exemplo JavaScript 28](#_Toc212060974)

[Figura 11 – Código exemplo de React Native 30](#_Toc212060975)

[Figura 12 - QR Code gerado pelo EXPO 32](#_Toc212060976)

[Figura 13 - Resultado da página de login 32](#_Toc212060977)

[Figura 14 – Código exemplo CSS 33](#_Toc212060978)

[Figura 15 – Resultado da página de login 35](#_Toc212060979)

[Figura 16 – Código exemplo Node.JS 36](#_Toc212060980)

[Figura 17 - Principais comandos NPM 37](#_Toc212060981)

[Figura 18 - Código exemplo Python 39](#_Toc212060982)

[Figura 19 - Resultado Exemplo Python 39](#_Toc212060983)

[Figura 20 - Wireframe de Baixa Fidelidade 41](#_Toc212060984)

[Figura 21 - Wireframe de Alta Fidelidade 42](#_Toc212060985)

[Figura 22 - Exemplo de diagrama de caso de uso 44](#_Toc212060986)

[Figura 23 - Exemplo de diagrama de caso de uso 46](#_Toc212060987)

[Figura 24 - Exemplo de diagrama de atividade 47](#_Toc212060988)

[Figura 25 - Exemplo de diagrama de Sequência 48](#_Toc212060989)

[Figura 26 – Exemplo de diagrama de Máquina de Estados 49](#_Toc212060990)

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

Automatic Gain Control (AGC)

Cascading Style Sheet (CSS)

Centrum Wiskunde & Informatica (CWI)

Corrente Contínua (CC)

European Computer Manufacturers Association (ECMA)

HyperText Markup Language (HTML)

Internet of Things (IoT)  
Língua Brasileira de Sinais (Libras)  
Organic Light Emitting Diode (OLED)  
Transcription Lens (TLens)  
Unified Modeling Language (UML)  
Universidade de São Paulo (USP)  
Sociedade Brasileira de Computação (SBC)  
Sociedade Brasileira de Fonoaudiologia (SBFA)

JavaScript Object Notation (JSON)

Node Package Manager (NPM)

Over-the-air (OTA)

Platform as a Service (PaaS)

Shell Script (SH)

Wireless Fidelity (Wi-Fi)

SUMÁRIO

[1 Introdução 14](#_Toc212150755)

[2 Referencial Teórico 16](#_Toc212150756)

[2.1 Deficientes Auditivos no Brasil 16](#_Toc212150757)

[2.2 Dificuldade na Comunicação Entre Deficientes Auditivos e Ouvintes 17](#_Toc212150758)

[2.3 Tecnologias de Desenvolvimento de Software 17](#_Toc212150759)

[2.3.1 JavaScript 17](#_Toc212150760)

[2.3.2 React Native 19](#_Toc212150761)

[2.3.3 Expo 21](#_Toc212150762)

[2.3.4 Cascading Style Sheet (CSS) 23](#_Toc212150763)

[2.3.5 Node.js 26](#_Toc212150764)

[2.3.6 Node Package Manager (NPM) 27](#_Toc212150765)

[2.3.7 Python 28](#_Toc212150766)

[2.3.8 Package Installer for Python (PIP) 30](#_Toc212150767)

[2.3.9 Vosk 30](#_Toc212150768)

[2.3.10 Mongo DB 30](#_Toc212150769)

[2.4 Arquitetura do Sistema e Infraestrutura Tecnológica 31](#_Toc212150770)

[2.4.1 Internet das Coisas (IoT) 31](#_Toc212150771)

[2.4.2 Orange Pi Zero 2W 31](#_Toc212150772)

[2.4.3 Módulo de Exibição OLED I2C 32](#_Toc212150773)

[2.4.4 Lentes Semirrefletivas Divisoras de Feixe 33](#_Toc212150774)

[2.4.5 Bateria de Íons de Lítio (BLI) 34](#_Toc212150775)

[2.4.6 Módulo Controlador de Carga 35](#_Toc212150776)

[2.4.7 Módulo Regulador de Tensão Step-Up 36](#_Toc212150777)

[2.4.8 Módulo de Microfone AGC Amplificador 37](#_Toc212150778)

[2.4.9 Wireless Fidelity (Wi-Fi) 37](#_Toc212150779)

[2.4.10 Ubuntu Server 38](#_Toc212150780)

[2.4.11 Shell Script (SH) 38](#_Toc212150781)

[2.4.12 Docker 40](#_Toc212150782)

[2.4.13 Container 40](#_Toc212150783)

[2.5 Prototipagem e Design do Sistema 40](#_Toc212150784)

[2.5.1 Modelagem 3D 40](#_Toc212150785)

[2.5.2 Wireframes 41](#_Toc212150786)

[2.5.3 Wireframes de Baixa Fidelidade 41](#_Toc212150787)

[2.5.4 Wireframes de Alta Fidelidade 42](#_Toc212150788)

[2.5.5 Figma 43](#_Toc212150789)

[2.5.6 User Experience (UX) 43](#_Toc212150790)

[2.5.7 User Interface (UI) 44](#_Toc212150791)

[2.5.8 Realidade Aumentada 44](#_Toc212150792)

[2.6 Modelagem e Documentação do Sistema 44](#_Toc212150793)

[2.6.1 Unified Modeling Language (UML) 44](#_Toc212150794)

[2.6.2 Diagrama de Casos de Uso 45](#_Toc212150795)

[2.6.3 Documentação de Casos de Uso 46](#_Toc212150796)

[2.6.4 Diagrama de Atividade 48](#_Toc212150797)

[2.6.5 Diagrama de Sequência. 49](#_Toc212150798)

[2.6.6 Diagrama de Máquina de Estados 50](#_Toc212150799)

[3 Desenvolvimento 52](#_Toc212150800)

[3.1 Diagrama de Caso de Uso 52](#_Toc212150801)

[3.1.1 Diagrama de Caso de Uso 52](#_Toc212150802)

[REFERÊNCIAS 58](#_Toc212150803)

# Introdução

O TLens um IoT acessível integrado a um par de óculos, que capta a fala e projeta sua transcrição diretamente nas lentes, desenvolvido com React (Meta, 2013) e microcontroladores programados para transcrição local, possibilitará ao usuário e responsáveis facilitar e monitorar a experiência de uso. A aplicação, em sua simplicidade e eficiência, é uma tecnologia diferencial em acessibilidade, proporcionando uma comunicação melhor, autônoma e fácil. Além disso, ele é voltado a pessoas com perda auditiva total ou parcial, abrangendo todas as faixas etárias, que enfrentam barreiras comunicacionais decorrentes da deficiência auditiva.

O estudo é apresentado na perspectiva de que a necessidade de ultrapassar as barreiras comunicacionais vivenciadas por pessoas com deficiência auditiva se faz imprescindível. Embora as tecnologias assistivas tenham avançado significativamente, as soluções de transcrição de fala são inacessíveis para a maioria da população por diversos motivos, principalmente financeiros, além da complexidade dos sistemas existentes.

Segundo a Sociedade Brasileira de Fonoaudiologia (2023), a deficiência auditiva interfere no processo de compreensão da fala, tornando os portadores do déficit a necessitar de estratégias alternativas para a compreensão explicativa da fala. Dessa forma, supõe-se que o uso do TLens possa contribuir significativamente para a inclusão social, com o acesso facilitado à fala em texto.

A problemática gira em torno de aparelhos auditivos de alto custo disponíveis no mercado e a impossibilidade de entender a fala de quem não domina a Língua Brasileira de Sinais (Libras). Além disso, muitos dispositivos atuais apresentam limitações técnicas e dependem de configurações complexas, que limita a interação social e profissional para grande parte da população com deficiência auditiva.

Decorrente disso, ocorre a exclusão social de pessoas com deficiência auditiva em função da barreira linguística. É muito difícil implementar políticas eficazes em um país onde a Libras não é comumente utilizada, o que potencializa o isolamento desses grupos. Afirma-se que a comunicação entre surdos e ouvintes, muitas vezes, depende da mediação de terceiros, comprometendo a espontaneidade das relações interpessoais. Souza e Mello (2021) destacam que uma aplicação capaz de projetar transcrições pode não apenas reduzir essa dependência, mas também aumentar a autonomia dos usuários e melhorar sua qualidade de vida.

Por esse motivo, o TLens tem como objetivo principal a criação de um sistema que possa ser vestível e baseado em IoT, com o intuito de mediar a comunicação entre surdos e ouvintes de um modo prático, com custo-benefício e que seja acessível, democratizando o uso da tecnologia. Para alcançar esse objetivo, foram estabelecidos os seguintes objetivos específicos:

1. Desenvolver um sistema que projete legendas diretamente nas lentes dos óculos.
2. Integrar um microfone unidirecional para converter fala em texto.
3. Desenvolver um aplicativo móvel para controlar e operar o dispositivo.
4. Criar um protótipo de baixo custo.

Hoje em dia, com o avanço acelerado da Internet das Coisas (IoT), a integração de objetos corriqueiros com sistemas inteligentes tem ocorrido cada vez mais. De acordo com Santos (2018), vivemos uma era em que nossos dispositivos estão cada vez mais inteligentes e conectados para exercer maior impacto na vida das pessoas. Em acessibilidade, um sistema IoT como o TLens representa um avanço para a comunicação, permitindo que pessoas com surdez interajam de forma mais independente em contextos sociais, educacionais e profissionais.

Neste documento serão abordadas as tecnologias aplicadas para a realização do sistema: React para o desenvolvimento do aplicativo de gerenciamento; BirdBath Optics e OLED para a parte óptica do sistema de projeção; C++ para a programação do microcontrolador responsável pelo reconhecimento e transcrição de fala; e UML 2 (Guedes, 2018) para a modelagem dos diagramas do sistema, como casos de uso, tipos de classes, classes e sequência.

# Referencial Teórico

Aqui veremos os principais fundamentos e tecnologias utilizados para formar a base teórica do projeto e o desenvolvimento do TLens.

## Deficientes Auditivos no Brasil

A perda de audição é uma condição que afeta significativamente a forma como uma pessoa se conecta e se relaciona. No Brasil, dados da Pesquisa Nacional de Saúde (PNS), feita pelo IBGE em 2019 e divulgada em 2021, apontam que em torno de 10,7 milhões de brasileiros possuem alguma deficiência auditiva, dos quais cerca de 2,3 milhões são enquadrados nos critérios legais de identificação da deficiência. Contudo, é bom lembrar que a maioria dessas pessoas não é composta por falantes da Língua Brasileira de Sinais (Libras). Isso apenas ressalta as particularidades que compõem o universo dos surdos e, por sua vez, a necessidade de políticas públicas voltadas para essa população, respeitando suas distintas manifestações de comunicação (IBGE, 2021).

Além das barreiras ligadas à comunicação, pessoas com deficiência auditiva também sofrem com a dificuldade de acessar serviços fundamentais, como saúde e educação. Um estudo conduzido por Oliveira et al. (2024) apontou que muitos profissionais da área da saúde não conseguem entender e atender de maneira adequada pacientes com deficiência auditiva, pois, em sua maioria, desconhecem a Libras. Tal barreira pode, inclusive, inviabilizar diagnósticos e tratamentos adequados.

No campo educacional, os desafios, de igual modo, são grandes. Embora tenham sido instituídas leis como a de nº 10.436/2002, que reconhecem a Libras como meio legal de comunicação, o que se observa no cotidiano das escolas não condiz necessariamente com aquilo que se espera de um verdadeiro ambiente inclusivo. A carência de formação específica para os professores, a falta de materiais didáticos desenhados para atender a esses estudantes e, ainda, a ausência de um corpo docente preparado para realmente atender às necessidades especiais deles, consistem em obstáculos ao seu adequado desenvolvimento escolar.

No mundo do trabalho, a questão não é muito diferente. Apesar da existência da Lei de Cotas (Lei nº 8.213/1991), que define a necessidade de empregar pessoas com deficiência, ainda são poucas as empresas que se encontram aptas a oferecer um local de trabalho acessível e confortável. Isso acaba por prejudicar a situação de quem tem deficiência auditiva, que se depara com poucas ou nenhumas propostas para progredir.

## Dificuldade na Comunicação Entre Deficientes Auditivos e Ouvintes

Como Stelling et al. (2014) salientam, a comunicação entre surdos e ouvintes merece atenção especial uma vez que esta está repleta de dificuldades causadas pela inexistência de uma língua comum. Muitas vezes, a falta do conhecimento da Língua Brasileira de Sinais (Libras) e da cultura surda por parte dos ouvintes que leva à exclusão e ao isolamento de pessoas surdas em vários contextos sociais. Está barreira idiomática impede a realização de um diálogo eficaz, prejudicando a interação e o desenvolvimento social, emocional e linguístico das pessoas surdas.

Segundo destacam Souza & Mello (2021), a comunicação é de extrema importância para a integração, a participação e a socialização do ser humano. Entretanto, as dificuldades de comunicação entre surdos e ouvintes aparecem quando não há um canal de linguagem partilhado, como a Libras. Esta situação pode significar que a pessoa surda terá um atraso no seu desenvolvimento e ainda este atraso pode afetar a sua participação plena no ambiente social e educativo.

## Tecnologias de Desenvolvimento de Software

Esta seção apresenta as linguagens, frameworks e bibliotecas aplicadas na construção do TLens.

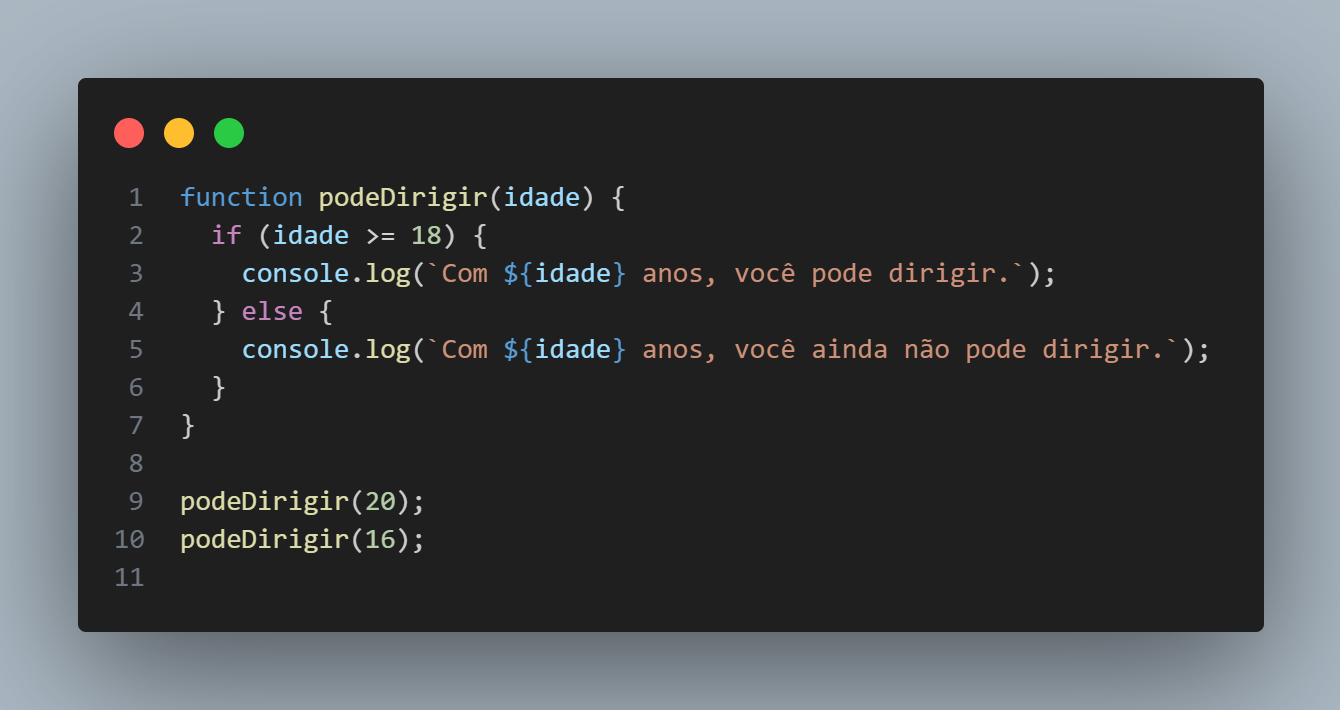
### JavaScript

De acordo com Lepesen (2018), JavaScript é uma linguagem de programação de 1995 criada pela Netscape e posteriormente padronizada pela *European Computer Manufacturers Association* (ECMA), para uso no navegador. Executada diretamente pelos navegadores, ela tornou-se uma ferramenta essencial para a Internet.

JavaScript é uma linguagem dinâmica, orientada a objeto e popularizada no desenvolvimento client-side. Grillo & Fortes (2008), JavaScript permite editar valores *HyperText Markup Language* (HTML), criar elementos, processar dados e verificar formulários.

Flanagan (2012) explica que o JavaScript é uma possível das linguagens de programação mais importantes da história da web, considerada, ao lado do HTML e do *Cascading Style Sheet* (CSS), parte da tríade da web.

Figura - Código Exemplo JavaScript



Fonte: Autoria Própria, 2025.

Na imagem acima, a função declarada em JavaScript verifica se uma determinada pessoa pode dirigir com base em sua idade. O texto abaixo descreve cada linha de código.

Linha 2: Declaração de uma função denominada de "podeDirigir", que recebe um parâmetro chamado idade. Esse valor será usado para verificar se a pessoa tem idade suficiente para dirigir.

Linha 3: Início de uma estrutura condicional if. Aqui, a função verifica se o valor da variável idade é maior ou igual a 18 (idade mínima para dirigir, conforme a legislação brasileira).

Linha 4: Caso a condição seja verdadeira (ou seja, se a pessoa tiver 18 anos ou mais), o código dentro deste bloco será executado. A função exibirá no console a mensagem informando que a pessoa pode dirigir, utilizando interpolação de string para incluir a idade na frase.

Linha 6: Caso a condição do if seja falsa (ou seja, a pessoa tiver menos de 18 anos), o bloco else será executado. Nesse caso, será exibida no console uma mensagem informando que a pessoa não pode dirigir, também incluindo a idade na frase.

Linha 9: Primeira chamada da função podeDirigir, passando o valor 20 como argumento. A função será executada e verificará se alguém com 20 anos pode dirigir.

Linha 10: Segunda chamada da função podeDirigir, agora com o valor 16. A função será executada novamente, verificando se alguém com 16 anos pode dirigir.

### React Native

O React Native foi introduzido publicamente em 2015 através de um anúncio do Facebook como afirmou Galvão (2018). A ferramenta já era usada para a criação de aplicativos IOS e posteriormente Android.

Conforme Falcão (2022) o React Native destaca-se entre as ferramentas de desenvolvimento multiplataforma oferecendo a possibilidade de construir aplicativos móveis com JavaScript, sem necessitar ter conhecimento profundo das linguagens nativas de cada sistema operacional.

Além disso, Galvão (2018) caracteriza um ponto forte do React Native dos demais é sua capacidade de proporcionar uma experiência próxima a nativa, isto é, utilizando componentes renderizados diretamente em ambos dos sistemas Android e IOS.

Figura – Código exemplo de React Native

Texto

O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria própria, 2025

Esse é um modelo de código básico do React Native que apresenta uma página de login básico com campos de email, senha e um botão de entrar. Detalhando mais o código:

Linha 1 a 3: Essas linhas são encarregadas de importar as dependências necessárias para o aplicativo. São trazidos componentes essenciais do React Native, como View, TextInput, TouchableOpacity, Text, Image, Alert e StyleSheet, além da função useState do React, utilizada para o controle de estado.

Linha 5: É iniciada a função principal da aplicação, LoginScreen, que representa a tela de login do app. Essa função é exportada como padrão (default) para que possa ser usada em outras partes do projeto.

Linha 6 a 7: São criadas as variáveis de estado email e password com seus respectivos atualizadores (setEmail e setPassword) por meio do hook useState, permitindo armazenar e modificar os dados digitados nos campos de entrada.

Linha 9 a 13: É definida a função handleLogin, responsável por verificar se os campos estão preenchidos. Caso estejam vazios, exibe um alerta de erro. Se ambos tiverem valor, exibe uma mensagem de boas-vindas com o email digitado.

Linha 15: Início do retorno JSX, que define os componentes visuais que serão renderizados na tela.

Linha 16: É utilizado o componente View como contêiner principal, onde todos os outros componentes da tela de login serão agrupados.

Linha 17 a 19: É adicionado um componente Image que exibe uma imagem (logo pequena do React Native), usando uma URL como fonte e aplicando estilos definidos em styles.logo.

Linha 20 a 24: Primeiro campo de entrada (TextInput) que captura o email do usuário. Define o tipo de teclado como "email", desativa a capitalização automática e liga o valor ao estado email, sendo atualizado a cada digitação.

Linha 25 a 29: Segundo campo de entrada (TextInput) para a senha, com o recurso secureTextEntry ativado para ocultar os caracteres. Também está conectado ao estado password.

Linha 30 a 32: Botão de login criado com TouchableOpacity. Ao ser pressionado, chama a função handleLogin. Dentro do botão, há um Text com o rótulo “Entrar”.

### Expo

O Expo é uma ferramenta que visa otimizar o desenvolvimento mobile com React Native através da remoção de etapas de configurações de ambientes nativos complicados. Segundo Falcão (2022) o Expo se destaca em automatizar processos de build e deploy de forma você não vai precisar de emuladores como Android Studio ou Xcode.

Conforme destaca Gomes (2023), além de facilitar o desenvolvimento, o Expo também se destaca por fornecer atualizações via *over-the-air* (OTA), recurso que permite a correção de bugs, dessa forma a manutenção fica mais ágil e eficiente até mesmo de aplicativos em produção.

Segundo Junior (2024), o Expo proporciona um ambiente de desenvolvimento que pode ser usado tanto em navegadores quanto em aparelhos físicos, por meio do aplicativo Expo Client, que está disponível na App Store e no Google Play.

Figura 12 - QR Code gerado pelo EXPO

Código QR

O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria própria, 2025

Depois de executar o comando "npm start" na pasta do projeto, a aplicação móvel começa a funcionar, gerando um código QR que deve ser lido por um dispositivo móvel para que a aplicação seja apresentada.

Figura - Resultado da página de login

Uma imagem contendo Interface gráfica do usuário

O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria própria, 2025

A figura acima representa o efeito da aplicação em um aparelho móvel sem qualquer estilização.

### Cascading Style Sheet (CSS)

O CSS (Cascading Style Sheets) é uma linguagem de estilo que permite estilizar documentos HTML ou XML. É importante mencionar que Jobstraibizer (2010) afirma que o CSS é um componente essencial para separar a estrutura do conteúdo da formatação, obtendo maior flexibilidade no desenvolvimento de uma página.

De certa forma o CSS tornou-se uma tecnologia essencial desde o momento em que ela foi criada. Conforme Silva (2010), a linguagem se tornou popular devido à sua capacidade de padronizar estilos e separar a camada de apresentação da estrutura dos documentos HTML.

Figura – Código exemplo CSS

Texto

O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria própria, 2025.

A seguir, apresentamos o código de estilização em React Native, que utiliza diversas propriedades inspiradas no CSS tradicional. Essas propriedades são essenciais para definir o layout e a aparência visual. A estrutura e função de cada linha são descritas abaixo:

Linha 1: Define uma constante styles e a inicializa com o objeto retornado por StyleSheet.create. Este objeto conterá todos os estilos que serão aplicados aos componentes da tela de login.

Linha 2: Inicia a definição de um estilo chamado container.  
Linha 3: flex: 1 – Faz com que o componente View ocupe todo o espaço disponível na tela.  
Linha 4: justifyContent: 'center' – Centraliza os elementos filhos verticalmente dentro do contêiner.  
Linha 5: alignItems: 'center' – Centraliza os elementos filhos horizontalmente no contêiner.  
Linha 6: backgroundColor: '#e6f0ff' – Define a cor de fundo do contêiner como azul-claro.

Linha 7: Inicia a definição de um estilo chamado logo.  
Linha 8: width: 120 – Define a largura da imagem como 120 pixels.  
Linha 9: height: 120 – Define a altura da imagem como 120 pixels.  
Linha 10: marginBottom: 30 – Adiciona uma margem inferior de 30 pixels abaixo da imagem, separando-a dos campos de entrada.

Linha 11: Inicia a definição de um estilo chamado input.  
Linha 12: width: '80%' – Faz com que o campo de entrada ocupe 80% da largura disponível.  
Linha 13: height: 45 – Define a altura do campo de entrada como 45 pixels.  
Linha 14: backgroundColor: '#fff' – Define a cor de fundo do campo como branco.  
Linha 15: borderRadius: 8 – Arredonda os cantos da borda em 8 pixels.  
Linha 16: paddingHorizontal: 15 – Adiciona um preenchimento interno horizontal de 15 pixels (esquerda e direita).  
Linha 17: marginBottom: 15 – Adiciona uma margem inferior de 15 pixels, separando este campo dos outros elementos.  
Linha 18: borderWidth: 1 – Define a largura da borda como 1 pixel.  
Linha 19: borderColor: '#a3b1c6' – Define a cor da borda como um cinza-azulado claro.

Linha 20: Inicia a definição de um estilo chamado button.  
Linha 21: width: '80%' – Faz com que o botão ocupe 80% da largura disponível.  
Linha 22: height: 45 – Define a altura do botão como 45 pixels.  
Linha 23: backgroundColor: '#3366ff' – Define a cor de fundo do botão como azul.  
Linha 24: borderRadius: 8 – Arredonda os cantos da borda do botão em 8 pixels.  
Linha 25: justifyContent: 'center' – Centraliza o conteúdo do botão verticalmente.  
Linha 26: alignItems: 'center' – Centraliza o conteúdo do botão horizontalmente.  
Linha 27: marginTop: 10 – Adiciona uma margem superior de 10 pixels, separando o botão dos outros elementos.

Linha 28: Inicia a definição de um estilo chamado buttonText.  
Linha 29: color: '#fff' – Define a cor do texto do botão como branco.  
Linha 30: fontSize: 18 – Define o tamanho da fonte do texto como 18 pixels.  
Linha 31: fontWeight: 'bold' – Define o peso da fonte como negrito.

Linha 32: Finaliza a definição do objeto StyleSheet.create que contém todos os estilos usados na tela de login.

Figura – Resultado da página de login

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria Própria, 2025.

A figura acima representa o efeito da aplicação em um aparelho móvel já com as estilizações aplicadas.

### Node.js

De acordo com Pereira (2013) o Node.js foi criado em 2009 com o objetivo de solucionar os problemas relacionados ao tráfego intenso de rede, usando um modelo inovador de arquitetura *non-blocking*, leve e eficiente (Moraes, 2018).

Segundo Moraes (2018) o Node.js também possibilita escrever o servidor e programar a lógica da aplicação usando apena Javascript. Além disso, pode ser usado tanto em sites quando em aplicativos, ou até mesmo desenvolver ferramentas de linha de comando (Pereira, 2013).

Figura – Código exemplo Node.JS

Texto

O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria própria, 2025.

O exemplo acima é um servidor simples criado com Node.js, utilizando o módulo nativo http, e configurado para rodar na porta 3000. Ao ser executado retorna a mensagem “Olá mundo!”. Abaixo a explicação linha por linha.

Linha 1: Importa o módulo nativo http, que permite a criação de servidores web.

Linha 3: Com a função “createServer”, o servidor é criado.

Linha 4: Define o cabeçalho da resposta que será enviado ao cliente, nesse exemplo será retornado texto.

Linha 5: Envia a mensagem “Olá, mundo! “ ao cliente.

Linha 6: Fecha a função da criação do servidor.

Linha 8: Define a porta 3000 onde o servidor executará as requisições.

Linha 9: Exibe no console uma mensagem informando que o servidor está em execução e por onde pode ser acessado.

Linha 10: Fecha a função.

### Node Package Manager (NPM)

Segundo Pereira (2013), o NPM é uma ferramenta de gerenciamento de pacotes nativa do Node.js. Além de possibilitar o compartilhamento de módulos entre desenvolvedores, é utilizado por linha de comando (Moraes, 2018). Entre os principais comandos estão:

Figura 17 - Principais comandos NPM

Texto

O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria Própria, 2025.

### Python

Segundo Borges (2014), a linguagem de programação Python foi criada em 1990 por Guido van Rossum, no Instituto Nacional de Pesquisa em Matemática e Ciência da Computação da Holanda (CWI). Foi criada com foco em usuários matemáticos.

De acordo com Menezes (2019), Python é uma linguagem poderosa, com sintaxe clara e objetiva. Ademais, Python é uma linguagem de programação associada à produtividade, pois nela existem muitas bibliotecas que podem ser usadas para fazer grandes coisas com pouco código.

Além disso, o Python possui código aberto, o que significa que pode ser usado de forma gratuita por programadores para desenvolver e distribuir um software, segundo Banin (2018).

Figura - Código exemplo Python

Texto

Descrição gerada automaticamente

Fonte: Autoria Própria, 2025.

O exemplo acima é uma função simples feita em Python, que realiza o cálculo da área de um retângulo. A seguir, uma breve explicação do código.

Linha 1: Declara a função chamada “calcular\_area\_retangulo”, que tem como objetivo fazer o cálculo da área do retângulo.

Linha 2 a 3: Foram declaradas duas variáveis, sendo elas “largura” e “altura”, que armazenarão o que o usuário digitar.

Linha 5: Foi declarada uma variável chamada “área”, que realiza o cálculo da área do retângulo.

Linha 7: Exibe na tela o resultado do cálculo.

Linha 9: Realiza a chamada da função “calcular\_area\_retangulo”.

Figura - Resultado Exemplo Python

Interface gráfica do usuário, Texto

O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria Própria, 2025.

A imagem acima ilustra o resultado após a chamada da função “calcular\_area\_retangulo”. Foram passados a altura e a largura, e, após isso, foi realizado o cálculo da área do retângulo.

### Package Installer for Python (PIP)

O PIP é uma ferramenta de gerenciamento de pacotes Python, podendo tanto instalar quanto desinstalar pacotes Python. O PIP também possui mensagens de erro mais compreensíveis e garante a instalação completa dos pacotes, evitando erros causados por instalações incompletas, segundo Reitz & Schlusser (2017).

### Vosk

Conforme apresentado por Cechim & Ganhor (2024) o Vosk é uma *toolkit* de reconhecimento de fala de código aberto e *offline*, desenvolvida para converter fala em texto. Ele se destaca por sua capacidade de operar sem conexão à internet, o que o torna ideal para dispositivos com recursos limitados, como Raspberry Pi e smartphones, bem como para aplicações que exigem baixa latência.

Parafraseando Lima et al. (2020), o Vosk, como uma biblioteca *offline* e de código aberto, oferece uma alternativa ao reconhecimento de fala. Diferente de serviços como Microsoft, Google e IBM, que utilizam APIs para processar áudio em servidores remotos, o Vosk realiza a transcrição da fala localmente no dispositivo, eliminando a necessidade de conectividade externa.

### Mongo DB

De acordo com Paniz (2023) o NoSQL se originou como uma resposta à necessidade emergente de novos modelos apropriados de armazenamento de dados para as novas necessidades aplicações da sociedade moderna. Impulsionando a importância de selecionar a ferramenta de acordo com necessidades que, a fim, podem ou não incluir bancos relacionais.

MongoDB é um banco de dados orientado a documentos que armazena as informações em estruturas no formato *JavaScript Object Notation* (JSON). Diferente dos documentos convencionais da família Microsoft. Os documentos no MongoDB contêm toda a informação necessária de forma autônoma, sem as restrições impostas pelos bancos de dados relacionais. Esse modelo favorece maior flexibilidade e agilidade no armazenamento e recuperação dos dados, especialmente em aplicações que lidam com grandes volumes de informações heterogêneas como afirma Boaglio (2020).

## Arquitetura do Sistema e Infraestrutura Tecnológica

Esta seção apresenta os componentes físicos e lógicos que sustentam o funcionamento do TLens, incluindo hardware, conectividade e sistemas operacionais utilizados.

### Internet das Coisas (IoT)

A Internet das Coisas (IoT) refere-se à interconexão digital entre os objetos físicos pela internet, ou seja, tornam-se dispositivos autônomos e eficientes capazes de comunicar-se entre si, não é necessário interação direta e constante de um ser humano para coletar, transmitir e processar dados tornando-se ativos em diversos sistemas (Santos, 2019). Utilizando tal tecnologia, é viável criar sistemas inteligentes e completos, auxiliando na melhoria de comunicação e interação em múltiplos domínios.

Por conseguinte, a IoT tem se posicionado de destaque ao possibilitar a integração de sensores, dispositivos ópticos e sistemas embarcados para criar soluções inovadoras às demandas de acessibilidade e inclusão social exigidos.

A rápida evolução das ferramentas digitais tem remodelado setores sociais e econômicos, destacando a IoT como uma das tecnologias com maior impacto, especialmente no desenvolvimento de dispositivos acessíveis que ampliam a comunicação e interação social (Silva et al., 2020).

### Orange Pi Zero 2W

A Orange Pi é uma empresa voltada ao desenvolvimento de hardware e software de código aberto, oferecendo placas acessíveis e versáteis. Esses dispositivos, segundo Pedamkar (2022 **apud** Wong Na, 2022), possuem conectividade Wi-Fi e são compatíveis com diferentes sistemas operacionais, o que permite sua aplicação em diversos tipos de projetos.

A Orange Pi Zero 2W é uma placa de desenvolvimento compacta e eficiente, ideal para projetos onde o hardware tem de ser o mais reduzido possível, mas que se pede uma performance aceitável. Com o processador Allwinner H618 a placa consegue ser superior à Raspberry Pi Zero 2W, que é a sua inspiração para seu design. Conta com o Wi-Fi nas bandas 2.4 GHz e 5 GHz, além de Bluetooth 5.0, aumentando sua comunicação com outros dispositivos. Disponível nas versões com até 4GB de memória RAM DDR4, consegue suportar sistemas operacionais (SO) como Armbian e Ubuntu sem problemas com travamentos. Apesar da ausência de slot para SSD, o uso de cartão microSD atende às necessidades de armazenamento. (Silva, 2025, p. 18-19). Abaixo temos um exemplo de uma placa Orange PI Zero 2W.

Figura - Orange Pi Zero 2W

Circuito eletrônico com fios

O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria Própria, 2025.

### Módulo de Exibição OLED I2C

O módulo de exibição OLED com módulos de comunicação I2C integrados são muito utilizados em sistemas com foco em IoT pela sua eficiência, baixo consumo e facilidade de implementação com microcontroladores. Eles operam com apenas dois pinos (SDA e SCL). Esse módulo permite a troca de dados com dispositivos de forma rápida e organizada, utilizando endereços específicos no barramento, segundo Costa (2024). Segue abaixo uma ilustração do módulo.

Figura - Módulo de Exibição OLED I2C



Fonte: Autoria Própria, 2025.

### Lentes Semirrefletivas Divisoras de Feixe

Um divisor de feixe, como a placa de vidro com 50% de transmitância e 50% de refletância, opera com base nos princípios fundamentais da interação da luz com a matéria. O fenômeno da reflexão, por exemplo, ocorre quando o raio luminoso incide em uma superfície e retorna ao mesmo meio de propagação (Gonzaga, 2022, p. 11). Paralelamente, ocorre a refração, que é o processo em que o raio de luz, ao incidir sobre uma interface, passa de um meio para outro.

A teoria que explica o funcionamento de superfícies semirrefletivas, essenciais para divisores de feixe, envolve a manipulação da luz através de revestimentos especializados. Quando a luz incide sobre uma superfície plana, parte dela é refletida no mesmo meio de incidência, enquanto outra parte é refratada, adentrando o segundo meio (INSTITUTO DE FÍSICA DE SÃO CARLOS (USP), 2015). Essa capacidade de dividir o feixe luminoso em proporções controladas é fundamental para a funcionalidade do componente.

Para se obter uma divisão específica, como 50% de transmitância e 50% de refletância, são empregadas técnicas de revestimento com filmes finos. Esses filmes, frequentemente dielétricos, são cuidadosamente projetados para controlar as propriedades ópticas da interface. A espessura e a composição desses filmes são cruciais, pois, como em outros contextos da física de materiais, "o estudo de camadas dielétricas" é realizado para compreender como a composição de filmes finos pode influenciar características desejadas em sistemas (Albertin, 2008). No caso dos divisores de feixe, essa tecnologia permite o balanço exato entre a luz transmitida e a refletida. A seguir um exemplo de lentes semirrefletivas divisoras de feixe.

Figura - Lentes Semirrefletivas Divisoras de Feixe



Fonte: Autoria Própria, 2025.

### Bateria de Íons de Lítio (BLI)

As Baterias de Íons de Lítio (BLIs) destacam-se pela alta densidade de energia, longa vida útil e baixa autodescarga, características que as tornam ideais para eletrônicos e veículos elétricos. Sua operação envolve o fluxo de íons entre cátodo e ânodo, conferindo-lhes alta eficiência. Apesar disso, desafios como custo e aspectos de segurança demandam sistemas de gerenciamento eficientes (Paula, 2024). Abaixo temos um exemplo Bateria de Li-ion.

Figura - Bateria de Li-ion

Interface gráfica do usuário, Texto

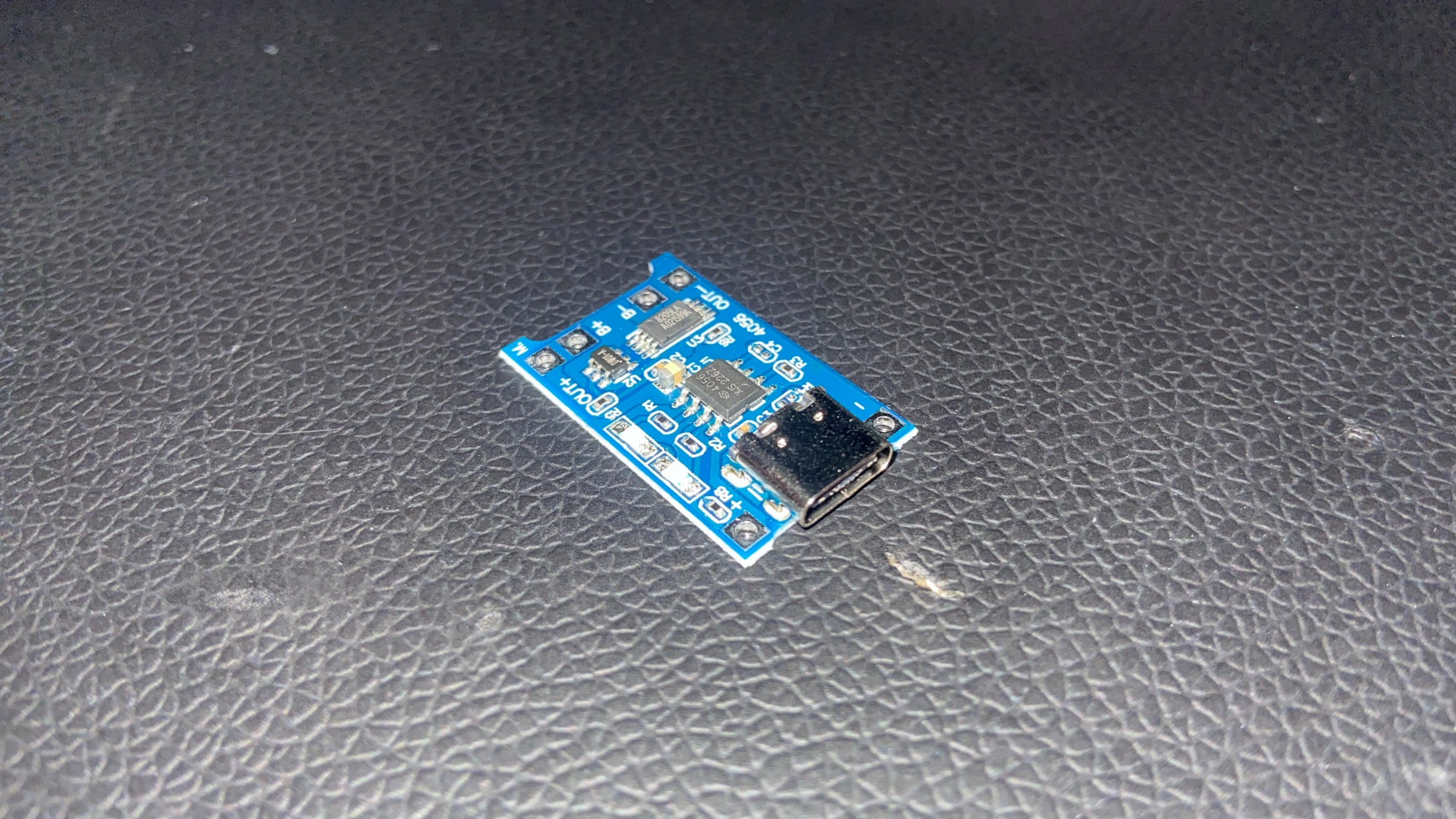
O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria Própria, 2025.

### Módulo Controlador de Carga

Um controlador de carga é um circuito essencial para prolongar a vida útil de baterias de lítio, atuando no controle dos limites máximo e mínimo de operação. Sua função primordial é prevenir a sobrecarga, que pode causar danos severos e até explosões, e a descarga total, que danifica as placas da bateria. Esse componente é vital em aplicações de Internet das Coisas (IoT), podendo ser recarregado via painel solar ou fonte de alimentação, garantindo a autonomia do dispositivo (Oliveira, 2021). Como pode ser observado na imagem abaixo um módulo controlador de carga.

Figura - Módulo Controlador de Carga

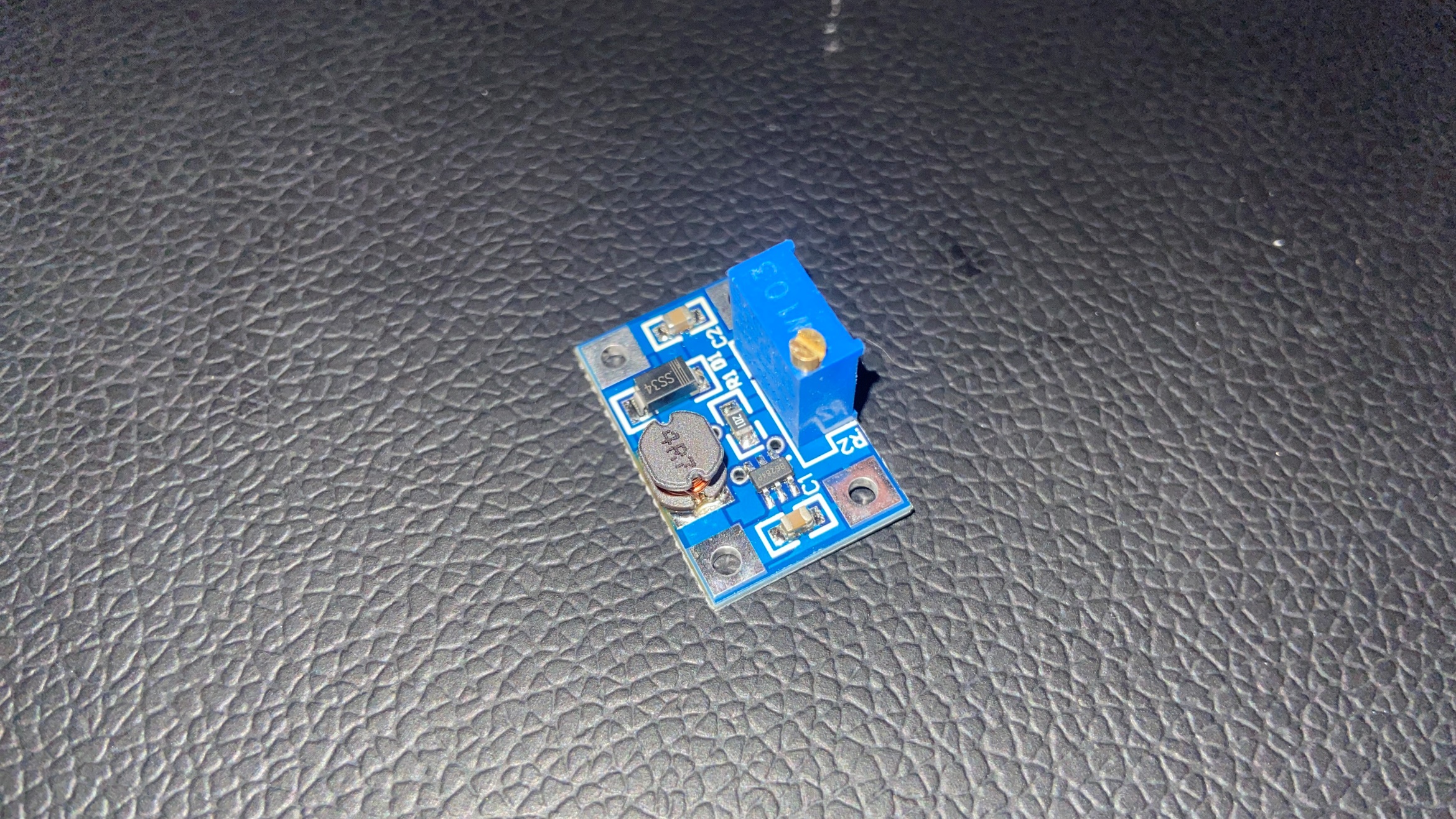


Fonte: Autoria Própria, 2025.

### Módulo Regulador de Tensão Step-Up

Um módulo regulador de tensão Step-Up, também conhecido como conversor CC-CC elevador de tensão, é um dispositivo projetado para aumentar o nível de tensão contínua (CC) de uma fonte de energia. Sua principal função é converter uma tensão de entrada mais baixa em uma tensão de saída mais elevada, sendo essencial em aplicações que necessitam de um suprimento de tensão superior ao disponível inicialmente (Piussi, 2017). Abaixo temos um exemplo de Módulo regulador de Tensão Step-Up.

Figura - Módulo Regulador de Tensão Step-Up

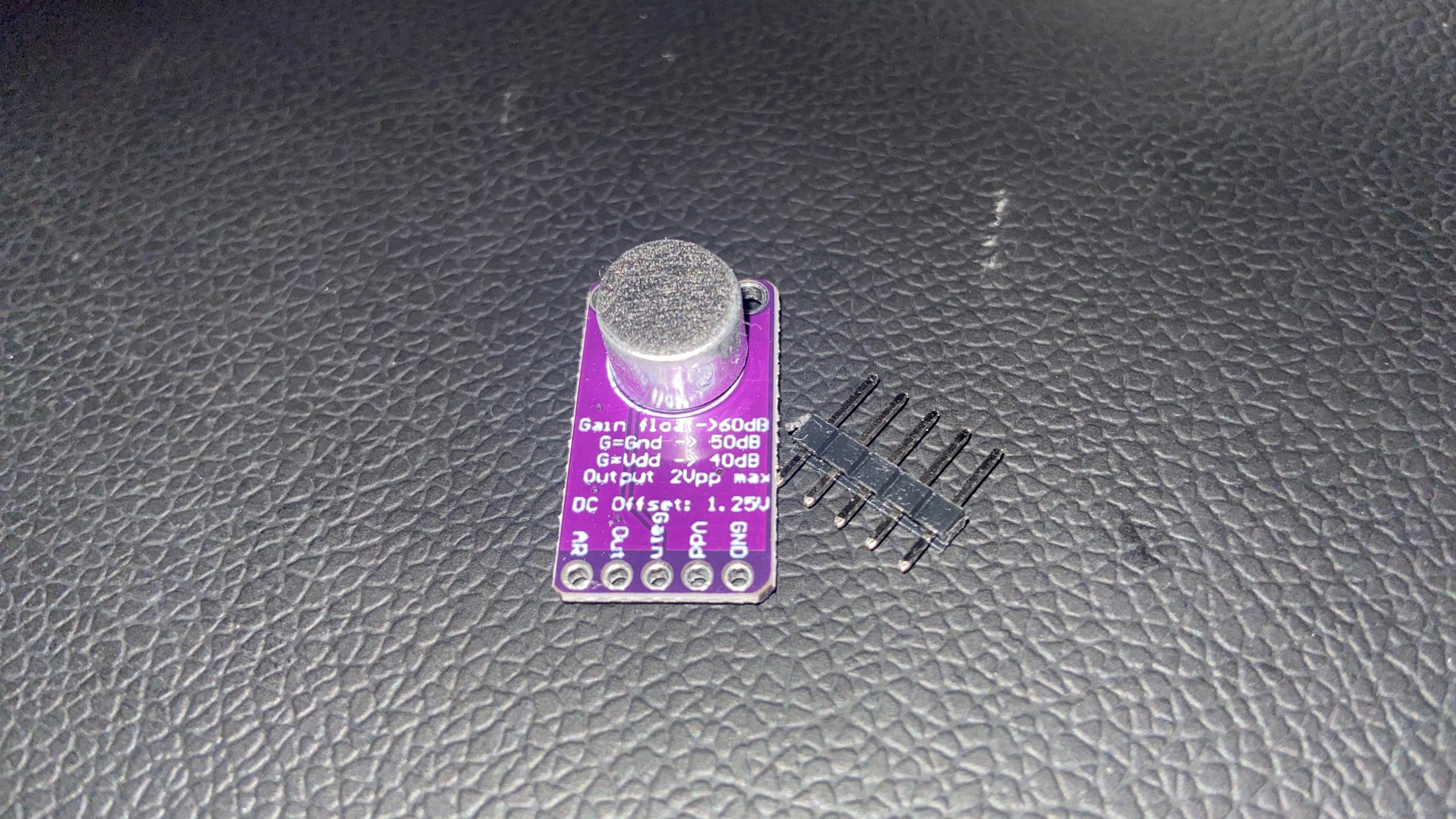


Fonte: Autoria Própria, 2025.

### Módulo de Microfone AGC Amplificador

Um Módulo de Microfone AGC Amplificador é um dispositivo que capta ondas sonoras e as amplifica, incorporando um Controle Automático de Ganho (AGC). Essa funcionalidade ajusta o sinal em tempo real, evitando distorções por sons muito altos e garantindo a clareza em volumes baixos. Esses módulos são valorizados pela sua sensibilidade, baixo custo e tamanho compacto, sendo ideais para diversas aplicações de captação sonora. (Leon, 2022). Veja a seguir uma ilustração de um módulo de microfone AGC amplificador.

Figura - Módulo de Microfone AGC Amplificador MAX9814



Fonte: Autoria Própria, 2025.

### Wireless Fidelity (Wi-Fi)

O Wireless Fidelity (Wi-Fi) surgiu com a aprovação do padrão IEEE 802.11 em 1997 promovendo comunicação sem fio entre os dispositivos. O IEEE “802.11b” tornou-se popular em 1999, oferecendo velocidade de até 11 Mbps e foi adicionado à Wi-Fi Alliance com a finalidade de garantir a compatibilidade entre os equipamentos (Museu Capixaba, 2023).

A utilização de redes públicas de Wi-Fi está submetida a vários problemas, principalmente os relacionados ao roubo de identificação que são ataques de engenharia social. Barros (2021) destaca que o Wi-Fi é uma tecnologia específica que funciona na transmissão sem fio em rede de computadores. No entanto, conectar-se a redes públicas Wi-Fi sem cuidados pode ajudar os hackers a roubar seus dados através de acessos aparelhos falsos

Ratusznei et al. (2019) analisaram a rede pública WiFi Livre SP como infraestrutura para cidades inteligentes. O objetivo foi avaliar a qualidade do serviço em aspectos como acesso, disponibilidade e desempenho. A pesquisa identificou falhas na estabilidade da conexão, recomendando melhorias técnicas para consolidar a rede como base eficiente de conectividade urbana. A engenharia Social se torna um jeito eficaz de captar informações confidenciais sem que a vítima sinta, alertando que há uma necessidade urgente no segurança de práticas nessas redes.

### Ubuntu Server

Um sistema operacional é o *software* essencial que permite ao usuário e aos programas interagirem com o *hardware* do computador. Ele tanto oferece uma interface simplificada para o uso dos recursos físicos, quanto gerencia esses mesmos recursos de forma eficiente, segundo Tanenbaum (2016).

Conforme Brito (2016), o Ubuntu Server, uma distribuição Linux da Canonical, idealizada por Mark Shuttleworth em 2004 e baseada no Debian, é uma plataforma versátil para diversos serviços e aplicações. Sua estabilidade, segurança e o vasto suporte da comunidade de código aberto o tornam uma escolha eficiente para gerenciar infraestruturas de rede, desde servidores web e de banco de dados até soluções em nuvem.

### Shell Script (SH)

Shell Script (SH) consiste em uma sequência de comandos executados ordenadamente pelo sistema, funcionando como um roteiro pré-definido de instruções e parâmetros. Essencialmente, ele agrupa comandos tipicamente inseridos manualmente em uma interface de linha, permitindo sua execução automatizada (Jargas, 2008).

Figura - Exemplo Shell Script

Texto

O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria própria, 2025

Esse código acima exemplifica alguns comandos SH, confira a explicação mais detalhada:

Linha 1: Conhecida como "shebang", esta linha deve ser sempre a primeira em um script shell. Ela indica ao sistema o interpretador a ser utilizado para a execução do código, neste caso, o bash.

Linhas 3 e 4: Apresentam comentários que visam descrever o propósito do código. Em Shell Script, todo comentário é precedido por um #, sinalizando ao interpretador que essa linha não deve ser processada como comando.

Linha 6: Exibe a mensagem "Olá, Mundo!" na tela, utilizando o comando echo, que imprime texto na saída padrão. As aspas duplas são empregadas para delimitar o conteúdo como uma *string*.

Linha 9: Imprime na tela a data e hora atuais do sistema. O comando date é inserido dentro de $(...) (estrutura conhecida como substituição de comando) para que o interpretador o execute e insira seu resultado na *string*.

Linha 12: Armazena a *string* "Usuário" na variável NOME.

Linha 13: Imprime a *string* "Olá, Usuário!". O símbolo $ antecede o nome da variável ($NOME) para indicar ao interpretador que deve acessar e exibir o conteúdo armazenado nela.

### Docker

De acordo com Vitalin e Castro (2016), Docker foi desenvolvido em 2013 pela empresa dotCloud, que decidiu fazer um *open source* em Platform as a Service (PaaS), originando uma ferramenta voltada à virtualização leve através de containers. Em poucos meses, o Docker ganhou ampla adesão da comunidade e foi rapidamente adotado por grandes empresas de tecnologia.

A utilização e relevância do Docker se faz necessária para a simulação complexa de sistemas, como uma rede blockchain, por que possibilita de criar ambientes isolados e eficientes como afirma Morais et al. (2020).

### Container

O Docker destacou-se por padronizar o uso de containers, tornando-se amplamente reconhecido por essa inovação, a ponto de seu nome se tornar diretamente associado à própria tecnologia de containers como citou Jacinto (2022).

Vitalino e Castro (2016) nos apresenta que containers são unidades isolados para unir as aplicações juntamente com suas dependências, compartilhando o kernel do sistema operacional que os torna mais leves e eficientes que máquinas virtuais. Entretanto, essa propriedade oferece portabilidade e assegura consistência entre a execução de softwares em diferentes ambientes.

Já a utilização destes meios automatizados facilita a implantação rápida de ambientes, com menor consumo de recursos e maior escalabilidade de acordo com Morais et al. (2020).

## Prototipagem e Design do Sistema

Esta seção aborda o processo visual e funcional do TLens.

### Modelagem 3D

Modelagem Tridimensional (3D) é uma fase fundamental no processo de manufatura aditiva, já que formam a representação digital dos objetos que serão fabricados. Segundo Júnior & Vaz (2020), toda impressão 3D se constitui sobre esta noção digital da peça, sendo assim, atenção aos detalhes geométricos, proporções e funcionalidades inerentes forma feita em softwares especializados, desde ferramentas voltadas para iniciantes até plataformas profissionais.

Modelagem 3D é definida como processo de formação de estrutura digital que representa elementos do mundo real ou fictício em três dimensionamentos – altura, largura e profundidade (Lopes, 2023). Abaixo temos um exemplo de Modelagem 3D.

Figura - Modelo 3D Haste de Óculos

Imagem de vídeo game

O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria Própria, 2025.

### Wireframes

De acordo com Memória (2006), os wireframes são um rascunho do conteúdo de uma determinada tela, posicionando hierarquicamente os elementos e construindo a navegação. Eles são bastante importantes, pois representam a primeira etapa para a construção de um projeto.

### Wireframes de Baixa Fidelidade

Segundo Rodrigues (2017), os wireframes de baixa fidelidade são uma forma simples e rápida de se construir uma interface para apresentação ao cliente. Porém, não possuem tanta fidelidade com a versão final, podendo ser descartados com facilidade.

Figura - Wireframe de Baixa Fidelidade

Uma imagem contendo Forma

O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria Própria, 2025.

### Wireframes de Alta Fidelidade

Os wireframes de alta fidelidade são uma representação mais fiel do produto final, utilizando materiais que, preferencialmente, serão aplicados no produto definitivo (Rodrigues, 2017).

Figura - Wireframe de Alta Fidelidade

Interface gráfica do usuário, Site

O conteúdo gerado por IA pode estar incorreto.

Fonte: Autoria Própria, 2025.

### Figma

Figma é uma ferramenta de design gráfico no navegador e se a instalação no computador, como comenta Cadore (2021), essa plataforma permite a criação de layouts de maneira clara e objetiva, que a acaba ajudando no processo de trabalho do designer.

De acordo com Villain & Silveira (2023), o Figma foi criado com o objetivo de ser uma ferramenta web gratuita e online para a criação de interfaces e protótipos. essa viabilidade foi uma das grandes motivações que ajudou a popularização da ferramenta especialmente entre iniciantes e freelancers.

Oliveira (2022) aponta o sucesso das comunidades de usuários do Figma, destacando a contribuição ativamente com extensões, sistemas de design e diretrizes de estilo.

### User Experience (UX)

De acordo com Teixeira (2022), a Experiência do Usuário (UX) abrange a totalidade das interações e percepções de um indivíduo com um produto ou serviço. Esse campo do design visa criar vivências positivas e satisfatórias, englobando aspectos emocionais, sensoriais e cognitivos.

A UX garante que produtos atendam às necessidades do usuário, proporcionando uma vivência intuitiva e eficiente. Tal disciplina é crucial para reduzir frustrações, gerar lealdade à marca e impactar positivamente os resultados financeiros das organizações (Grant, 2019).

A aplicação do UX é direcionada pela coleta dos requisitos do usuário, que são as necessidades expressas por eles, distintas dos requisitos técnicos. Essa etapa é crucial, pois sem essa ferramenta, não é possível desenvolver um artefato alinhado à realidade, exigindo a audição e o teste com usuários reais para direcionar as atividades de design, utilizando técnicas como entrevistas e prototipagem (Stati; Sarmento, 2021).

### User Interface (UI)

A *User Interface* (UI) refere-se à porção visual de uma interface, englobando tanto a aparência dos elementos de interação e do conteúdo quanto a organização espacial e o posicionamento de todos os componentes na página ou tela. Seu principal propósito é ser projetada com o usuário em mente, visando otimizar a qualidade da interação com um dispositivo, produto ou serviço (Kruk, 2015, p. 10).

### Realidade Aumentada

Segundo Tori & Hounsell (2020), a realidade aumentada permite a integração de objetos reais com objetos gerados computacionalmente, como textos, imagens e objetos. Essa tecnologia mantém o campo de visão do usuário no mundo real ao mesmo tempo que adiciona elementos virtuais.

## Modelagem e Documentação do Sistema

Esta seção demonstra a modelagem e representação do sistema, utilizando diagramas UML para ilustrar sua estrutura, comportamento e interações funcionais.

### Unified Modeling Language (UML)

Segundo Guedes (2018), UML é uma linguagem de modelagem usada para prototipar softwares baseados no paradigma de orientação a objetos. Atualmente, ela é considerada a linguagem padrão para modelagem de software.

Consoante Booch, Rumbaugh & Jacobson (2012), a UML é bastante útil, pois, com ela, o desenvolvedor consegue tomar decisões importantes, como analisar e projetar um novo software.

Sendo uma linguagem de modelagem, o objetivo da UML é auxiliar os desenvolvedores a definirem as características do sistema, seus requisitos e seu comportamento. Com a UML, esses objetivos podem ser definidos antes mesmo de o software começar a ser desenvolvido (Guedes, 2018).

### Diagrama de Casos de Uso

Para Guedes (2018), o diagrama de casos de uso é o primeiro e mais simples diagrama. Ele apresenta uma linguagem simples para um entendimento geral e exibe os atores e as funcionalidades do sistema.

Ademais, esse diagrama contém elementos como atores e casos de uso. Atores são os diferentes papéis que os usuários dos casos de uso podem desempenhar, e casos de uso são as próprias funcionalidades do sistema, segundo Booch, Rumbaugh & Jacobson (2012).

Figura - Exemplo de diagrama de caso de uso

Diagrama

Descrição gerada automaticamente

Fonte: Guedes, 2018.

O exemplo abordado acima representa um simples diagrama de caso de uso que tem como principais elementos:

Ator: Nos diagramas de casos de uso, são representados por bonecos palitos. Eles são representações de usuários ou sistemas externos que interagem diretamente com o sistema.

Levantamento de Requisitos: São as principais funcionalidade que o sistema precisa ter, sendo eles requisitos funcionais, não funcionais e regras de negócios.

Casos de uso: Os casos de uso são representados por uma elipse e por um verbo no infinitivo. Eles são os próprios requisitos do sistema.

Associação: Tem como objetivo demonstrar as interações entre os atores e os casos de uso. São representados por uma reta.

Generalização: Conhecido também como herança, é usada para indicar que um elemento herda uma característica ou comportamento de outro. Pode ser utilizada entre atores ou entre casos de uso. É representada por uma reta com uma seta que aponta para o elemento do qual o outro herda.

Inclusão: Tem como objetivo demonstrar que uma ação é obrigatória dentro de um sistema. É representado por uma linha pontilhada com “<<Include>>” escrito em cima ou abaixo.

Extensão: Tem como objetivo demonstrar caminhos opcionais dentro de um sistema. Pode ser utilizado para demonstrar decisões ou comportamentos alternativos, que podem ou não ser executados. É representado por uma reta pontilhada com “<<extend>>” escrito em cima ou abaixo.

### Documentação de Casos de Uso

Na visão de Guedes (2018) a documentação de casos de uso descreve, de maneira muito simples, as principais características do caso de uso, como, por exemplo, quais funções devem ser executadas pelo sistema e quais atores interagem com ele. Além disso não existe um modelo específico para ser seguido sobre essa documentação. A UML propõe certos formatos, todavia, é bastante flexível, podendo ser adaptado como preferir.

Figura - Exemplo de diagrama de caso de uso

Tabela

Descrição gerada automaticamente com confiança média

Fonte: Guedes, 2018.

O exemplo acima demonstra uma documentação de caso de uso “Emitir saldo” onde fornecerá as seguintes informações:

Nome do caso de uso: esse nome deve identificar o objetivo do caso de uso.

Ator principal: Identifica o ator que mais interage com o caso de uso.

Atores secundários: Identifica os atores que interagem em menor quantidade com o caso de uso; sua declaração não é obrigatória.

Resumo: Tem como objetivo descrever rapidamente o caso de uso.

Pré-condições: Estabelece pré-condições, ou seja, condições que precisam ser verdadeiras para que o caso de uso ocorra.

Pós-condições: Descreve as pós-condições, ou, seja, etapas que precisam ser seguidas após o caso de uso ser executado.

Fluxo principal: Tem como objetivo enumerar as ações que normalmente são executadas pelo caso de uso.

Exemplo: Primeiro, o cliente informa o número da conta; então, o sistema verifica a existência da conta e solicita a senha. Em seguida, o cliente informa a senha, o sistema verifica se a senha está correta e, por fim, emite o saldo.

Fluxos alternativos: São caminhos que podem ser seguidos quando o fluxo principal não pode ser concluído.

Exemplo: se o sistema não achou a conta, ele pode comunicar o usuário que a conta não foi encontrada, encerrando assim do fluxo principal.

Restrições e validações: Têm por finalidade definir regras para tornar o sistema mais consistente.

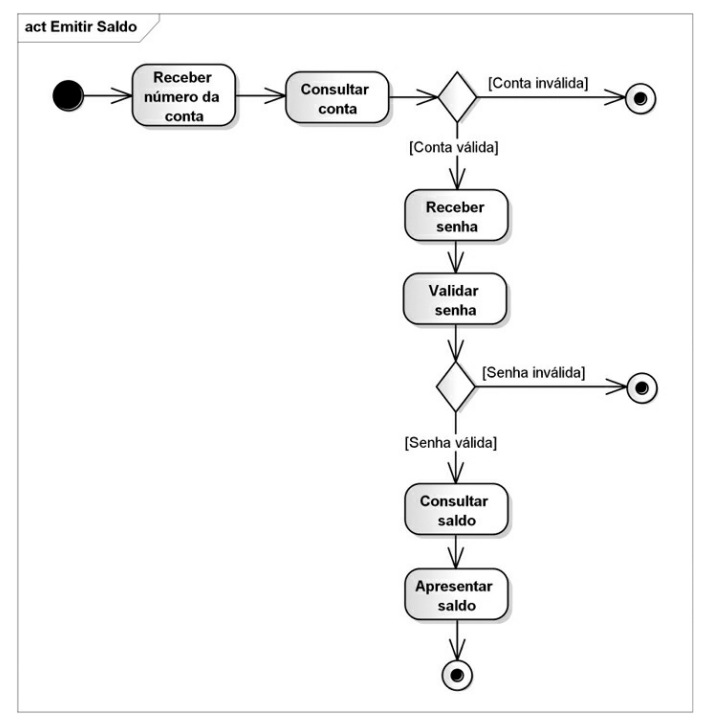
Por exemplo: o cliente precisa ser maior de idade para acessar o sistema.

### Diagrama de Atividade

Como afirma Booch, Rumbaugh e Jacobson (2012), os diagramas de atividade são usados para modelar as etapas sequenciais em um processo, também é usado para modelar o fluxo de um objeto.

Esse diagrama dá bastante ênfase ao nível de algoritmo da UML, ele destaca a sequência e condições para descrever comportamentos do sistema (Guedes, 2018).

Figura - Exemplo de diagrama de atividade



Fonte: Guedes, 2018.

No exemplo acima, é mostrado a atividade “Emitir Saldo” que é construída pela separação de processos após o nó de ação “Receber número da conta”. A seguir uma breve explicação do funcionamento desse diagrama.

O início da atividade “Emitir Saldo” é representado pelo nó de ação (círculo preto). Em seguida, ocorrem as primeiras atividades do sistema: “Receber número da conta”, e “Consultar conta” (retângulo grande com as bordas arredondadas).

Após isso, há uma decisão: se a conta for inválida, o fluxo é encerrado (círculo com contorno); se for válida, o fluxo continua.

A próxima atividade é “Receber senha”, seguida por “Validar senha”, que também possui uma decisão: se a senha for inválida, o fluxo é encerrado; se for válida o fluxo continua.

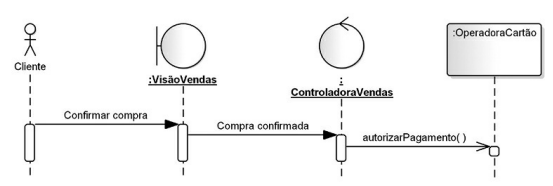
Logo após, ocorre a atividade “consultar saldo”, seguida de “apresentar saldo”. Por fim, o fluxo é encerrado pelo nó de atividade final.

### Diagrama de Sequência.

Segundo Guedes (2018) o diagrama de sequência é um diagrama comportamental que possui foco na sequência de eventos ocorridos em um processo. Ele determina como os objetos interagem dentro de um processo.

Ademias, ele é representado por uma linha tracejada vertical chamada “linha da vida”, que tem como objetivo mostrar a existência do objeto ao longo do tempo. Além disso, para indicar a passagem de tempo o fluxo ocorre de cima para baixo (Booch; Rumbaugh; Jacobson, 2012).

Figura - Exemplo de diagrama de Sequência



Fonte: Guedes, 2018.

Esse é um diagrama UML que representa o processo de confirmação de uma compra com cartão.

Temos o ator “Cliente”, que confirma a compra para a “VisãoVendas”. A interface envia a solicitação para a “CalculadoraVendas” que chama a função “autorizarPagamento()” na operadora do cartão, solicitando a autorização do pagamento.

### Diagrama de Máquina de Estados

Na visão de Guedes (2018) o diagrama de máquina de estado tem como objetivo

representar como um elemento muda de estado conforme alguns eventos ocorrem. Além disso, pode ser utilizado para representar o comportamento de partes individuais de um sistema.

Figura – Exemplo de diagrama de Máquina de Estados

Diagrama

Descrição gerada automaticamente

Fonte: Guedes, 2018.

Explicação do Diagrama acima.

Esse é um diagrama UML de máquina de estado que representa o processo de realização de um depósito bancário.

O processo inicia no estado inicial, representado pelo círculo preto.

Em seguida, o sistema entra no estado “ConsultandoConta”, onde é executada a ação “consultarConta()”. Essa ação serve para verificar se a conta é válida.

Se a conta for válida, o sistema passa para o estado “RecebendoValor”, onde aguarda o cliente informar o valor a ser depositado.

Depois disso, o sistema entra no estado “DepositandoValor”, executando a ação “depositarValor()” que realiza o depósito na conta.

Com o saldo atualizado, o sistema avança para o estado “RegistrarMovimento”, onde registra o movimento financeiro.

Por fim, o processo termina no estado final, representado pelo círculo com um ponto central.

# Desenvolvimento

Neste capítulo, serão apresentados e demonstrados os processos de desenvolvimento do TLens desde o seu planejamento até a sua realização. Começando pela parte teórica, sendo elas diagramas UML.............

## Diagrama de Caso de Uso

A seguir será demostrado o caso de uso do TLens, apresentando como o deficiente auditivo interage com o IoT dentro do sistema do TLens.

### Diagrama de Caso de Uso

No sistema TLens os requisitos funcionais correspondem as funcionalidades e serviços que o sistema deve oferecer, sendo executados a partir da interação do usuário. Geralmente esses requisitos servem de base para o desenvolvimento das telas e das operações que o sistema realizará.

Referindo-se a requisitos não funcionais, são aqueles que não está diretamente relacionado as funcionalidades do sistema, normalmente sendo utilizado para a parte de infraestrutura do sistema segurança, armazenamento e desempenho.

As regras de negócios refletem as políticas da organização, estabelecendo diretrizes que definem o que pode ou não ocorrer em determinadas situações.

**Requisitos funcionais:**

* Emparelhar os óculos no aplicativo
* Transcrever falas
* Exibir notificações no aplicativo
* Gerenciar histórico de transcrições
* Exibir o percentual da bateria dos óculos
* Verificar baixa porcentagem da carga da bateria
* Gerenciar tamanho, velocidade e lado de exibição (esquerda ou direita) da fonte
* Desligar os óculos

**Requisitos não funcionais**

* O sistema deve transcrever rapidamente as falas
* O emparelhamento entre os óculos e o app deve ser estável e persistente
* Os dados das conversas devem ser armazenados de forma segura

**Regras de negócio**

* A transcrição deve ser armazenada localmente quando os óculos estão emparelhados ao app, com opção de exclusão manual
* A transcrição pode acontecer sem o emparelhamento ao app
* O sistema deve atualizar o nível de bateria em tempo real, com precisão mínima de ±2%
* O sistema deve emitir um alerta ao usuário quando a carga estiver abaixo de 5%

Pode-se observar o caso de uso na figura abaixo, que demostra a interação entre o deficiente auditivo e o IoT dentro do sistema TLens com todas as suas funções respectivas, conforme diretrizes da UML. Além disso, segue a documentação dos casos de uso.

Colocar a imagem

Colocar a documebtaçãp cmdkckf foto

Tabela 1 – Descrição do caso de uso: Emparelhar Óculos

|  |  |
| --- | --- |
| Nome do Caso de Uso | Emparelhar Óculos |
| Ator Principal | TLens |
| Ator Secundário |  |
| Resumo | Este caso de uso descreve as etapas percorridas para o emparelhamento dos óculos ao aplicativo |
| Pré-condições |  |
| Pós-condições |  |
| Cenário Principal | |
| Ações do Ator | Ações do sistema |
| 1. Ligar os óculos |  |
|  | 2. Apresentar a lista de dispositivos TLens encontrados |
| 3. Escolher dispositivo para emparelhar |  |
|  | 4. Emparelhar dispositivo ao aplicativo |
| Cenário Alternativo - Dispositivo não encontrado | |
| Ações do Ator | Ações do sistema |
|  | 1. Informar que não foram encontrados dispositivos TLens |
|  |  |

Tabela 2 – Descrição do caso de uso: Gerenciar Histórico

|  |  |
| --- | --- |
| Nome do Caso de Uso | Gerenciar Histórico |
| Ator Principal | Deficiente Auditivo |
| Ator Secundário |  |
| Resumo | Este caso de uso descreve as etapas percorridas para o gerenciamento do histórico de falas. |
| Pré-condições | Efetuar o caso de uso "Emparelhar óculos" |
| Pós-condições |  |
| Cenário Principal | |
| Ações do Ator | Ações do sistema |
| 1. Acessar a opção “Conversas” no aplicativo. |  |
|  | 2. abrir a tela do histórico de transcrições, permitindo visualizar ou excluir transcrições |
| Cenário Exceção – Histórico Vazio | |
| Ações do Ator | Ações do sistema |
| 1. Acessar a opção “Conversas” no aplicativo. |  |
|  | 2. Detectar que não há transcrições salvas. |
|  | 3. Exibe a mensagem: "Nenhum histórico disponível no momento." |

Tabela 3 – Descrição do caso de uso: Gerenciar Fonte

|  |  |
| --- | --- |
| Nome do Caso de Uso | Gerenciar Fonte |
| Ator Principal | Deficiente Auditivo |
| Ator Secundário |  |
| Resumo | Este caso de uso descreve as etapas percorridas para o gerenciamento da fonte exibida nos óculos |
| Pré-condições | Efetuar o caso de uso "Emparelhar óculos" |
| Pós-condições |  |
| Cenário Principal | |
| Ações do Ator | Ações do sistema |
| 1. Acessar a opção “Conversas” no aplicativo. |  |
|  | 2. abrir a tela do histórico de transcrições, permitindo visualizar ou excluir transcrições |
| Cenário Exceção – Histórico Vazio | |
| Ações do Ator | Ações do sistema |
| 1. Acessar a opção "Meu TLens" no aplicativo |  |
|  | 2. abrir a tela de gerenciamento da fonte, permitindo troca de tamanho, velocidade e lado de visualização nos óculos (direita ou esquerda) |

Tabela 4 – Descrição do caso de uso: Gerenciar Brilho

|  |  |
| --- | --- |
| Nome do Caso de Uso | Gerenciar Brilho |
| Ator Principal | Deficiente Auditivo |
| Ator Secundário |  |
| Resumo | Este caso de uso descreve as etapas percorridas para o gerenciamento do brilho da fonte |
| Pré-condições |  |
| Pós-condições |  |
| Cenário Principal | |
| Ações do Ator | Ações do sistema |
| 1. Acionar o botão de configurar brilho (no app ou nos óculos) |  |
|  | 2. Identificar a origem do comando. |
| 3. Ajusta o nível de brilho desejado. |  |
|  | 4. Aplicar o novo nível de brilho. |

Tabela 5 – Descrição do caso de uso: Verificar bateria

|  |  |
| --- | --- |
| Nome do Caso de Uso | Verificar Bateria |
| Ator Principal | TLens |
| Ator Secundário |  |
| Resumo | Este caso de uso descreve as etapas percorridas para o gerenciamento da bateria |
| Pré-condições | Efetuar o caso de uso "Emparelhar óculos" |
| Pós-condições |  |
| Cenário Principal | |
| Ações do Ator | Ações do sistema |
|  | 1. Recuperar as informações sobre o nível de carga da bateria. |
|  | 2. O sistema exibe o nível atual da bateria e as opções disponíveis. |

Tabela 5 – Descrição do caso de uso: Gerenciar Notificações

|  |  |
| --- | --- |
| Nome do Caso de Uso | Gerenciar Notificações |
| Ator Principal | TLens |
| Ator Secundário |  |
| Resumo | Este caso de uso descreve as etapas percorridas para o gerenciamento das notificações |
| Pré-condições | Efetuar o caso de uso "Emparelhar óculos" |
| Pós-condições |  |
| Cenário Principal | |
| Ações do Ator | Ações do sistema |
| 1. Acessar a opção "Notificações" no aplicativo |  |
|  | 2. Enviar notificações |
| 3. Visualizar ou excluir notificações |  |

Tabela 6 – Descrição do caso de uso: Transcrever Falas

|  |  |
| --- | --- |
| Nome do Caso de Uso | Transcrever Falas |
| Ator Principal | TLens |
| Ator Secundário | Deficiente Auditivo |
| Resumo | Este caso de uso descreve as etapas percorridas para a transcrição de falas |
| Pré-condições | 1. Os óculos estar ligado  2. Microfone estar ligado e funcionando |
| Pós-condições |  |
| Cenário Principal | |
| Ações do Ator | Ações do sistema |
| 1. Iniciar a função de transcrição (pelo botão nos óculos ou no app). |  |
|  | 2. Capturar o áudio ambiente. |
|  | 3. Processar o áudio |
|  |  |
| Cenário Exceção – Histórico Vazio | |
| Ações do Ator | Ações do sistema |
| 1. Acessar a opção "Meu TLens" no aplicativo |  |
|  | 2. abrir a tela de gerenciamento da fonte, permitindo troca de tamanho, velocidade e lado de visualização nos óculos (direita ou esquerda) |

# REFERÊNCIAS

AGÊNCIA BRASIL. **Inclusão de surdos no mercado de trabalho ainda enfrenta desafios.** 2023. Disponível em: https://agenciabrasil.ebc.com.br/es/node/1378404. Acesso em: 18 maio 2025.

ALBERTIN, Kátia Franklin. **Estudo de camadas dielétricas para aplicação em capacitores MOS**. 2008. Tese (Doutorado em Engenharia Elétrica) – Universidade de São Paulo, São Paulo, 2008. Disponível em: https://www.teses.usp.br/teses/disponiveis/3/3140/tde-08012008-144158/publico/Tese\_Katia\_Franklin\_Albertin.pdf. Acesso em: 23 maio 2025.

BANIN, Sérgio Luiz. **Python 3 Conceitos e Aplicações:** Uma Abordagem Didática. 1. ed. São Paulo: Saraiva Educação, 2018.

BARROS, Emanoel Guilherme. **Acesso sem controle à internet: uma abordagem com engenharia social através de wireless fidelity (Wi-Fi).** Monumenta, Paraíso do Norte, PR, v. 2, n. 1, p. 59-63, mar. 2021.

BARROS, Emanoel Guilherme. **Acesso sem controle à internet: uma abordagem com engenharia social através de wireless fidelity (Wi-Fi).** Monumenta, Paraíso do Norte, PR, v. 2, n. 1, p. 59-63, mar. 2021.

BOAGLIO, Fernando. MongoDB: **Construa novas aplicações com novas tecnologias.** São Paulo, SP: Casa do Código, 2020. ISBN: 978-85-5519-043-8

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. **UML:** guia do usuário. 2. ed. Rio de Janeiro: Elsevier, 2012.

BORGES, Luiz Eduardo. **Python para Desenvolvedores:** Aborda Python 3.3. 1. ed. São Paulo: Novatec Editora Ltda, 2014.

BRASIL. Instituto Brasileiro de Geografia e Estatística. Pesquisa Nacional de Saúde: 2019: **percepção do estado de saúde, estilos de vida, doenças crônicas e saúde bucal.** Rio de Janeiro: IBGE, 2021. Disponível em: https://www.ibge.gov.br Acesso em: 18 maio 2025.

BRASIL. Lei nº 10.436, de 24 de abril de 2002. Dispõe sobre a Língua Brasileira de Sinais - Libras. **Diário Oficial da União:** seção 1, Brasília, DF, 25 abr. 2002. Disponível em: http://www.planalto.gov.br/ccivil\_03/leis/2002/L10436.htm. Acesso em: 18 maio 2025.

BRASIL. Lei nº 8.213, de 24 de julho de 1991. Dispõe sobre os Planos de Benefícios da Previdência Social. **Diário Oficial da União:** seção 1, Brasília, DF, 25 jul. 1991. Disponível em: http://www.planalto.gov.br/ccivil\_03/leis/l8213cons.htm. Acesso em: 18 maio 2025.

BRITO, Samuel Henrique Bucke. **Serviços de Redes em Servidores Linux**. São Paulo: Novatec, 2016.

CADORE, Douglas Cristiano. **UX DESIGN: um estudo sobre o desenvolvimento de produtos e serviços focado na experiência dos usuários.** 2021. 72 f. TCC (Graduação) - Curso de Administração, Universidade de Caxias do Sul, Guaporé, 2021. Cap. 60. Disponível em: https://repositorio.ucs.br/xmlui/handle/11338/9383. Acesso em: 18 maio 2025.

CECHIM, Filipe Potrich; GANHOR, João Paulo. **Omnis Chess:** Desenvolvimento de um xadrez adaptado para Pessoas Com Deficiências Físicas dos Membros Superiores por Meio de Hardware e Software Livre. In: CONGRESSO LATINO-AMERICANO DE SOFTWARE LIVRE E TECNOLOGIAS ABERTAS (LATINOWARE), 21., 2024, Foz do Iguaçu. **Anais...** Foz do Iguaçu: Latin.Science, 2024. p. 1-10. Disponível em: https://sol.sbc.org.br/index.php/latinoware/article/view/31580. Acesso em: 5 maio 2025.

COSTA, Matheus Queiroz. **Desenvolvimento de Dispositivo Portátil e Wireless para Análise Colorimétrica RGB de Concentração de Substâncias**. 2024. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Elétrica) – Universidade Federal de Uberlândia, Uberlândia, 2024. Disponível em: https://repositorio.ufu.br/bitstream/123456789/42243/1/DesenvolvimentoDispositivoPort%C3%A1til.pdf. Acesso em: 5 maio 2025.

CUNHA, Atacílio Costa. **Localização de dispositivos móveis usando roteadores com antenas direcionais e classificação de dados**. 2014. 75 f. Dissertação (Mestrado em Informática) — Instituto de Ciências Exatas, Universidade Federal do Amazonas, Manaus, 2014.

FALCÃO, Francisco Daniel. **Desenvolvimento de uma aplicação web para o gerenciamento de recursos computacionais.** 2022. 55 f. Trabalho de Conclusão de Curso (Graduação em Engenharia de Computação) – Universidade Federal do Ceará, Campus de Sobral, Sobral, 2022. Disponível em: https://repositorio.ufc.br/bitstream/riufc/69029/3/2022\_tcc\_fdfalc%C3%A3o.pdf. Acesso em: 25 maio 2025.

FALCÃO, Francisco Davi. **Turistando Beberibe: desenvolvimento de aplicativo com React Native e Expo**. 2022. 64 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Universidade Federal do Ceará, Fortaleza, 2022. Disponível em: https://repositorio.ufc.br/bitstream/riufc/69029/3/2022\_tcc\_fdfalc%C3%A3o.pdf. Acesso em: 25 maio 2025.

FLANAGAN, David. **JavaScript – O Guia Definitivo.** 6. ed. Porto Alegre: Bookman, 2012. 1080 p.

GALVÃO, Pedro Sereno. **Comprehensive Repository Analysis of Mobile Projects Built with Reactive Native.** 2018. 28 f. Dissertação (Programa de Graduação em Ciência da Computação) – Universidade Federal de Pernambuco, Recife, 2018.

GONZAGA, Fabíola. **UNIVERSIDADE ESTADUAL DA PARAÍBA CAMPUS VIII CENTRO DE CIÊNCIA, TECNOLOGIA E SAÚDE DEPARTAMENTO DE FÍSICA CURSO DE LICENCIAT**. João Pessoa: UEPB, 2022. Disponível em: https://dspace.bc.uepb.edu.br/jspui/bitstream/123456789/28738/4/TCC%20-%20Fabiola%20Gonzaga.pdf. Acesso em: 23 maio 2025.

GRANT, Will. **UX Design:** Guia definitivo com as melhores práticas de UX. São Paulo: Novatec, 2019.

GRILLO, Filipe del Nero e FORTES, Renata Pontin de Mattos. **Aprendendo javascript**. São Carlos: ICMC-USP. Disponível em: https://repositorio.usp.br/directbitstream/4cd7f9b7-7144-40f4-bfd0-7a1d9a6bd748/nd\_72.pdf. Acesso em: 23 maio 2025.

GUEDES, Gilleanes. **UML 2:** Uma Abordagem Prática. 3. ed. São Paulo: Novatec Editora Ltda., 2018.

JACINTO, Ulisses dos Santos. **Blindagem de Ambientes Virtuais Baseados em Contêineres: Docker Hardening**. 2022. 52 f. Monografia (Programa de Graduação em Ciência da Computação) – Universidade Federal do Maranhão, São Luís, 2022.

JARGAS, Aurelio Marinho. **Shell Script Profissional.** São Paulo: Novatec, 2008.

JOBSTRAIBIZER, Flávia. **Criação de sites com o CSS.** São Paulo: Érica, 2010.

JÚNIOR, Edison dos Passos Neri; VAZ, Cristina Lúcia Dias. **Guia de Impressão 3D.** ed. Belém: EditAedi/UFPA, 2020.

KRUK, Maryana. **User Interface de aplicações móveis para museus:** atenção partilhada. 2015. 119 f. Dissertação (Mestrado em Design de Comunicação e Novos Media) – Faculdade de Belas-Artes, Universidade de Lisboa, Lisboa, 2015. Disponível em: https://repositorio.ulisboa.pt/bitstream/10451/24074/2/ULFBA\_TES\_907.pdf. Acesso em: 25 maio 2025.

LEON, Giuliana Oliveira de Mattos. **AusculSensor:** um sistema para auxílio na ausculta pulmonar. 2022. 135 f. Trabalho de Conclusão de Curso (Graduação em Engenharia de Computação) – Universidade Federal do Pampa, Bagé, 2022.

LEPSEN, Edécio Fernando. **Lógica de Programação e Algoritmos com JavaScript**: uma Introdução à Programação de Computadores com Exemplos e Exercícios Para Iniciantes. 2. ed. São Paulo: Novatec Editora Ltda, 2022. 352 p.

LIMA, Matheus N. S. M. de et al. **Ferramentas e recursos disponíveis para reconhecimento de fala em Português Brasileiro.** In: COMPUTER ON THE BEACH, 10., 2020, Balneário Camboriú. **Anais...**. Balneário Camboriú: Editora da Univali, 2020. p. 1-6. Disponível em: https://periodicos.univali.br/index.php/acotb/article/view/17441. Acesso em: 25 maio 2025.

LOPES, Michele. **Modelagem 3D: o que é e como funciona.** Ebac Online, 29 de setembro de 2023. Disponível em: https://ebaconline.com.br/blog/modelagem-3d-o-que-e-e-como-funciona/.

MASSOLA, Silze Cristina; PINTO, Giuliano Scombatti. **O Uso da Internet das Coisas (IOT) a Favor da Saúde**. 2018. 14 f. Faculdade de Tecnologia de Taquaritinga (FATEC), São Paulo, 2018.

MEMÓRIA, Felipe. **Design Para a Internet:** Projetando a experiência perfeita. 111 ed. São Paulo Elsevier Editora Ltda, 2006.

MENEZES, Nilo Ney Coutivo. **Introdução à Programação Com Python.** 3. ed. São Paulo: Novatec Editora Ltda, 2019.

MEYER, Eric A. **CSS:** a linguagem da web. 3. ed. Rio de Janeiro: Alta Books, 2010. Disponível em: https://books.google.com.br/books?id=\_HBVQ-w5ZcoC. Acesso em: 26 maio 2025.

MORAES, William Bruno**. Construindo Aplicações com NodeJS.** 3. ed. São Paulo: Novatec Editora Ltda, 2022.

MORAIS, Anderson Melo de; LINS, Fernando Antonio Aires; CALLOU, Gustavo Rau de Almeida. **Simulação e Avaliação de Desempenho de uma Rede Blockchain Utilizando Containers Docker**. Cadernos do IME - Série Informática, v. 44, p. 74-81, jul. 2020.

MUSEU CAPIXABA**.** **Redes sem fio: IEEE 802.11 (Wi-Fi) de 1997.** Museu Capixaba de Ciência e Tecnologia, 2023. Disponível em: https://museucapixaba.com.br/hoje/redes-sem-fio-ieee-802-11-wifi-de-1997. Acesso em: 26 maio 2025.

OLIVEIRA, George Moreno de. **DESENVOLVIMENTO E AVALIAÇÃO DO PLUGIN PARA O FIGMA PARA DOCUMENTAÇÃO DE ACESSIBILIDADE PARA INTERFACES - DAI**. 2022. 79 f. TCC (Graduação) - Curso de Design Digital, Campus de Quixadá, Universidade Federal do Ceará, Quixadá, 2022. Disponível em: http://repositorio.ufc.br/handle/riufc/65589. Acesso em: 18 maio 2025.

OLIVEIRA, Sérgio. **Internet das coisas com ESP8266, Arduino e Raspberry Pi**. 2. ed. São Paulo: Novatec, 2021.

OLIVEIRA, Thais S. de; et al. **A inclusão de deficientes auditivos no sistema público de saúde.** Revista Foco, v. 11, n. 1, p. 57-66, 2024. Disponível em: https://ojs.focopublicacoes.com.br/foco/article/view/6791. Acesso em: 18 maio 2025.

PANIZ, David. NoSQL: **Como armazenar os dados de uma aplicação moderna**. São Paulo, SP: Casa do Código, 2023. ISBN:978-85-5519-192-3

PAULA, Abner Fernandes de. **Análise da utilização de baterias de íons de lítio em equipamentos de movimentação logística**: uma perspectiva técnica e financeira. 2024. 54 f. Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica) – Faculdade de Engenharia, Universidade Federal de Juiz de Fora, Juiz de Fora, 2024.

PEREIRA, Caio Ribeiro. **Aplicações web real-time com Nodejs.** 1. ed. São Paulo: Casa do Código, 2014.

PIUSSI, Alexandre Dellamora. **Projeto e implementação de um conversor CC-CC elevador de tensão para uma bancada didática de eletrônica de potência e controle digital de sistemas**. 2017. 113 f. Monografia (Graduação em Engenharia de Controle e Automação) – Universidade Federal de Santa Maria, Santa Maria, RS, 2017.

RATUSZNEI, Juliano et al. **Uma rede Wi-Fi aberta de larga escala como infraestrutura para cidades inteligentes**. In: SIBGRAPI – Conference on Graphics, Patterns and Images, 2019.

RODRIGUES, Paulo Henrique de Araujo. **Aplicação do Conceito Visual Material Design no Desenvolvimento de um Protótipo de Interface Gráfica.** 2017. 51 f. Trabalho de Conclusão de Curso (Graduação) – Tecnologia em Análise e Desenvolvimento de Sistemas. Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2017.

SANTOS, Sandro. **Introdução à IoT: Desvendando a Internet das Coisas**. 1. ed. Santa Catarina: Clube de Autores, 2019.

SILVA, Ávilla I. S. et al. **Segurança em aplicações de Internet das Coisas:** Bluetooth Low Energy, casos de uso e vulnerabilidades. In: LIVRO DE MINICURSOS SBRT, 2020.

SILVA, José Klinsman Jorge. **Air CoPilot**: um copiloto artificial para auxiliar os pilotos de drones e prevenir acidentes. 2025. 66 f. Trabalho de Conclusão de Curso (Tecnólogo em Análise e Desenvolvimento de Sistemas) – Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, Cajazeiras, 2025. Capítulo: “Placa Orange Pi Zero 2W”, p. 18-19.

SILVA, Paulo Henrique da**.** **Desenvolvimento de Aplicações com React Native e Expo**. 2022. 54 f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Universidade Federal Rural do Semi-Árido, Angicos, 2022. Disponível em: https://repositorio.ufersa.edu.br/bitstreams/c3533e38-0594-4d98-b7aa-d210b4ebaa2c/download. Acesso em: 25 maio 2025.

SOUZA, Andre Luis A. de; MELLO, Evelyn Bernadino. **As barreiras na comunicação de surdos com ouvintes**: uma reflexão sobre os papéis de cada indivíduo. Revelli, v. 13, 2021. ISSN 1984-6576. Disponível em: https://www.revista.ueg.br/index.php/revelli/article/view/10241. Acesso em: 29 abr. 2025.

STATI, Cesar Ricardo; SARMENTO, Camila Freitas. **Experiência do usuário (UX).** Curitiba: InterSaberes, 2021.

STELLING, Esmeralda Peçanha et al. **Pais ouvintes e filho surdo**: dificuldades de comunicação e necessidade de orientação familiar. Revista Espaço, Rio de Janeiro, n. 42, p. 15–25, jul./dez. 2014.

TANENBAUM, Andrew S.; BOS, Herbert. **Sistemas Operacionais Modernos**. 4. ed. Tradução de Daniel Vieira e Jorge Ritter. Revisão técnica de Raphael Y. de Camargo. São Paulo: Pearson Education do Brasil, 2016.

TEIXEIRA, Fabrício. **Introdução e Boas Práticas em UX Design.** 3. ed. São Paulo: Casa do Código: Alura, 2022.

UNIVERSIDADE DE SÃO PAULO. Instituto de Física de São Carlos. **Laboratório de Óptica**. São Carlos, 2015.

VILLAIN, Mateus; SILVEIRA, Maria Isabelle. **Figma: o que é a ferramenta, design e uso. o que é a ferramenta, Design e uso.** 2023. Disponível em: https://www.alura.com.br/artigos/figma. Acesso em: 18 maio. 2025.

VITALINO, Jeferson Fernando Noronha; CASTRO, Marcus André Nunes. **Descomplicando o Docker**. Rio de Janeiro: Brasport, 2016.

WONG NA, Marissa. **Emulação de IED embarcando o protocolo MMS em um sistema computacional de baixo custo**. 2022. 65 f. Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica) – Universidade Federal Fluminense, Escola de Engenharia, Niterói, RJ, 2022. Disponível em: http://app.uff.br/riuff/handle/1/27329. Acesso em: 21 maio 2025.