CartNinja: solução integrada entre *IoT*, *serverless* e computação em nuvem para auxiliar na redução do desperdício residencial

CartNinja: an integrated solution combining IoT, serverless, and cloud computing to help reduce houlsehold waste

Engenharia de Computação

Gabriel Lara Baptista *(pro15969@cefsa.edu.br)*
Mestre em Engenharia de Produção pela Universidade Nove de Julho e professor da Faculdade Engenheiro Salvador Arena

Caio Ruiz *(caio.ruiz00@gmail.com)*
Bacharel em Engenharia de Computação pela Faculdade Engenheiro Salvador Arena

Fernando Zanardi Lopes *(fzanardilopes@gmail.com)*
Bacharel em Engenharia de Computação pela Faculdade Engenheiro Salvador Arena

Thales Feliciano Baracho *(thalesfeliciano@hotmail.com)*
Bacharel em Engenharia de Computação pela Faculdade Engenheiro Salvador Arena

Victor Assagra *(victorassagra@hotmail.com)*
Bacharel em Engenharia de Computação pela Faculdade Engenheiro Salvador Arena

FTT Journal of Engineering and Business
• SÃO BERNARDO DO CAMPO, SP JUN. 2025
• ISSN 2525-8729
Submissão: 13 dez. 2024 Aceitação: 28 maio 2025
Sistema de avaliação: às cegas dupla (double blind review)
FACULDADE ENGENHEIRO SALVADOR ARENA, p. 76 - 95

FTT JOURNAL of Engineering and Business

FACULDADE ENGENHEIRO SALVADOR ARENA

Resumo

Enquanto milhões de pessoas sofrem de algum grau de insegurança alimentar, bilhões de toneladas de alimentos são desperdiçados anualmente. Diante desta perspectiva, tem-se que, no Brasil, aproximadamente 60% do desperdício advém de residências. Este projeto visa auxiliar na diminuição do desperdício de alimentos, sobretudo no ambiente residencial, através de uma solução que torne mais fácil e eficiente a criação de uma lista de compras, promovendo o consumo responsável. Fazendo uso da tecnologia de *IoT*, *serverless* e computação em nuvem, juntamente com um aplicativo móvel e um hardware dedicado com Raspberry Pi 3B+, a solução proposta criou um ambiente facilitado para o usuário gerenciar sua lista de compras. A ideia de uma lista de compras virtual surgiu a partir da pesquisa bibliográfica de artigos que mostraram a existência desse número alarmante de desperdício de alimentos em residências, sendo um dos motivos a existência de itens duplicados em despensas. A solução técnica desenvolvida se mostrou eficiente em testes realizados em três residências diferentes, de modo que houve relatos de melhoria na gestão das compras de produtos ausentes na despensa.

Palavras-chave: Internet das Coisas. Serverless. Computação em nuvem. Lista de compras virtual.

Abstract

While millions of people suffer from some degree of food insecurity, billions of tons of food are wasted annually. In this context, it is noted that in Brazil, approximately 60% of food waste comes from households. This project aims to help reduce food waste, especially in the residential environment, through a solution that makes it easier and more efficient to create a shopping list, promoting responsible consumption. Using IoT technology, serverless computing, and cloud computing, along with a mobile app and dedicated hardware with a Raspberry Pi 3B+, the proposed solution created an easier environment for users to manage their shopping list. The idea of a virtual shopping list came from the literature review of articles that showed this alarming number of food waste in homes, one of the reasons being duplicated pantry items. The technical solution developed proved efficient in tests conducted in three different households, with reports of improvements in managing the purchase of missing pantry products.

Keywords: Internet of Things. Serverless. Cloud Computing.Virtual Shopping List.

Introdução

De acordo com dados do Instituto Brasileiro de Geografia e Estatística (IBGE, 2019), aproximadamente 10,3 milhões de brasileiros vivem em condições de severa privação alimentar. Entre os 68,9 milhões de domicílios do Brasil, 36,7% enfrentam algum grau de insegurança alimentar, afetando diariamente 84,9 milhões de pessoas. Essa realidade evidencia um desafio alimentar para a atual geração.

Em contraponto a esse fato, enfrenta-se um paradoxo alarmante: enquanto, anualmente, bilhões de toneladas de comida são desperdiçadas, a fome e a insegurança alimentar persistem como desafios globais. A Organização das Nações Unidas (ONU, 2023) afirma que apesar de haver recursos suficientes para alimentar toda a população mundial, a distribuição inadequada perpetua a desigualdade alimentar.

Notadamente, o Brasil destaca-se como sendo o 10º país que mais desperdiça alimentos no mundo, com um total de 27 milhões de toneladas de alimentos desperdiçados por ano, conforme relatório da ONU (2021). Vale destacar que as famílias lideram esse desperdício de alimentos, sendo responsáveis por 61% deste desperdício, de acordo com informações levantadas pelo Programa das Nações Unidas para o Meio Ambiente, Pnuma (2021).

A partir desse cenário, percebe-se que as estatísticas apontam que as pessoas frequentemente têm dificuldade em gerenciar seus estoques residenciais, adquirindo alimentos duplicados e muitas vezes não percebendo os itens prestes a vencer, o que resulta em desperdício (Farr-Wharton *et al.*, 2014). Diante dessa situação urgente, é necessário adotar abordagens inovadoras que não apenas otimizem o uso dos recursos, mas que também abordem diretamente uma das causas fundamentais, mas oculta, da insegurança alimentar e da fome: o desperdício.

Partindo dessa problemática do desperdício de alimentos, este projeto surgiu como uma resposta direta a essa demanda premente. O objetivo geral não foi apenas melhorar a gestão do tempo das pessoas durante suas atividades de compras, mas também auxiliar no combate ao desperdício alimentar ao oferecer uma solução que minimize o problema e que seja viável economicamente para que se alcance mais pessoas. Combinando a tecnologia da Internet das Coisas (do inglês *internet of things*) e um aplicativo móvel, buscou-se criar uma experiência de compra mais eficiente e consciente.

Com o intuito de atingir o objetivo deste estudo, foram estabelecidos os seguintes objetivos específicos. Em primeira instância, visou-se a identificação de produtos por meio da leitura de códigos de barras, utilizando um protótipo de hardware microcontrolado para integrar esses itens a um carrinho de compras virtual. Adicionalmente, pretendeu-se possibilitar que o usuário adicionasse produtos ao carrinho através de comandos de voz para itens sem códigos de barras. Além disso, foi prevista a criação de uma

infraestrutura em nuvem dedicada, destinada a hospedar e executar os serviços do *backend*, bem como gerenciar o banco de dados associado. Esses objetivos específicos foram delineados para orientar o desenvolvimento da solução, visando uma abordagem abrangente e funcional para sua implementação.

Referencial teórico

A integração de dispositivos eletrônicos tornou-se uma tendência significativa no cenário da automação residencial. A interconexão entre hardware e software, nesse contexto, visou proporcionar uma experiência mais eficiente e personalizada aos usuários, otimizando o gerenciamento de recursos domésticos e promovendo maior comodidade (Danzinger, 2023). Nesse sentido, para o entendimento deste projeto, foi fundamental explorar as bases teóricas que fornecem o embasamento necessário para o desenvolvimento e a implementação desses sistemas.

Ademais, as práticas adotadas e a ideia do projeto foram inspiradas em um artigo base, que serviu como guia fundamental para a condução do projeto. Para tal, foi pensado, além da ótica de controle de data de validade dos produtos, também o controle do estoque através das compras futuras, ou seja, o que o usuário realmente utilizou e do que vai necessitar em sua próxima compra. Outro diferencial foi acrescentar a leitura por código de barras em vez de utilizar visão computacional (Reis et al., 2023).

Hardware: Raspberry Pi 3 B+ e outros componentes

Para servir de hardware deste projeto, foi utilizado um Raspberry Pi 3 B+, um microfone INMP441, um MH-ET Live Scanner para leitura de código de barras, um *led*, um botão, responsável pela ativação das funções de reconhecimento de voz e um HC-05 para facilitar a conexão *bluetooth* com a aplicação *mobile*. Todo esse equipamento foi acomodado dentro de uma caixa feita de resina, construída com o auxílio de uma impressora 3D, que consiste em um dispositivo capaz de criar objetos tridimensionais a partir de modelos digitais (Basniak; Liziero, 2017).

A Raspberry Pi 3B+ é uma placa de computador de baixo custo e alto desempenho, projetada para uma variedade de aplicações em computação e eletrônica, equipada com um processador quad-core de 64 bits ARM Cortex-A53 rodando a 1,4 GHz, e 1 GB de RAM. Além disso, possui conectividade *wireless* integrada, incluindo Wi-Fi 802.11ac de banda dupla e *Bluetooth* 4.2, o que permitiu uma fácil conexão com redes locais e dispositivos externos. Com uma variedade de portas de entrada e de saída, como portas USB,

HDMI e GPIO, a Raspberry Pi 3 B+ ofereceu uma plataforma versátil para projetos de automação residencial, servidores de mídia, aprendizado de programação, entre outros (Raspberry Pi Foundation, 2024).

O INMP441 oferece uma solução compacta e de alta qualidade para a captura de áudio em dispositivos eletrônicos, sendo um microfone digital de eletreto com saída PDM (Modulação por Densidade de Pulso). Essa saída o torna eficiente na transmissão de dados de áudio digital, tornando-o especialmente adequado para aplicações em que a eficiência do processamento e a qualidade de áudio são essenciais, como em dispositivos *IoT*, sistemas de reconhecimento de voz e outros produtos eletrônicos que demandam captação de som com alta fidelidade (Von Pflug; Krischker, 2017).

No contexto deste projeto, o microfone INMP441 utilizou a interface I2S para transmitir os dados de áudio digital de forma eficiente e com alta fidelidade. A interface de áudio I2S (*Inter-IC Sound*) é um protocolo serial de alta velocidade amplamente utilizado para transmitir áudio digital entre dispositivos em sistemas embarcados. Essa interface é adequada para aplicações que demandam qualidade de áudio superior e eficiência no processamento, tornando-a uma escolha ideal para sistemas de reconhecimento de voz e outras aplicações de captura de áudio em dispositivos *IoT* (Prajwal; Sowmya, 2022).

Para este projeto foi considerada a leitura de códigos de barras do tipo EAN-13, um padrão de 13 (treze) dígitos amplamente utilizados em mercados e lojas, sendo o último dígito um verificador e os primeiros dois ou três dígitos, identificadores do país de origem (Siaw-Yeboah, 2022).

Computação em nuvem

A computação em nuvem refere-se a um modelo de computação que oferece recursos de processamento, armazenamento e serviços através da internet. Em vez de manter servidores e infraestrutura local, os recursos são disponibilizados por provedores de serviços em *data centers* remotos, permitindo o acesso flexível e escalável a esses recursos sob demanda (Google Cloud, 2024). De acordo com AWS (2024), *IoT* ou Internet das Coisas refere-se a uma rede interconectada de dispositivos e a uma tecnologia que viabiliza a comunicação tanto entre esses dispositivos como entre a nuvem.

Dentro deste contexto, a Amazon Web Services (AWS) oferece uma gama diversificada de serviços de nuvem desenhados para atender diferentes necessidades de computação, armazenamento e comunicação de empresas e desenvolvedores. Para tal, foi necessária a criação de APIs em nuvem, que

servem para integrar os serviços; a sigla API (*application programming interface)* refere-se a um conjunto de normas, ferramentas e protocolos que possibilitam a interação de diferentes plataformas e softwares (AWS, 2024). Para a construção das APIs, foi utilizado o padrão REST, que é um estilo de arquitetura baseado no protocolo HTTP, utilizado na construção de sistemas com interfaces bem definidas. Ele estabelece conexões entre o cliente e o servidor de destino que podem gerar excelentes resultados em aplicações *mobile*, web e Internet das Coisas (IBM, 2024).

Para este projeto foram utilizados os serviços AWS, dentre eles: o *Lambda,* um serviço de computação sem servidor, que permite aos desenvolvedores executarem código em resposta a eventos, sem se preocupar com a infraestrutura, escolhido pelo baixo custo e alta disponibilidade.
O serviço Amazon API *Gateway* facilita a implementação de APIs e foi escolhido para expor as funções do AWS *Lambda* para a internet. Já o Amazon DynamoDB, um banco de dados NoSQL gerenciado, foi escolhido pelo baixo custo e alta disponibilidade. Por fim, foi também utilizado o Amazon SNS, um serviço de mensagens gerenciado, utilizado destinado ao envio de *e-mails* com a senha de uso único para os clientes, escolhido pela simplicidade do uso (AWS, 2024).

Docker

O Docker é uma plataforma de código aberto que simplifica a implantação e execução de aplicativos em contêineres, fornecendo uma maneira eficiente e padronizada de criar, distribuir e executar aplicativos, isolando-os uns dos outros e do sistema operacional subjacente. Essa abordagem facilita a escalabilidade, a implantação rápida e a consistência nas implementações de software. (Docker Docs, 2024)

Linguagens e frameworks

As linguagens de programação e *frameworks* são ferramentas fundamentais no desenvolvimento do software, cada uma com suas próprias características, vantagens e aplicações específicas. As linguagens de programação oferecem os blocos de construção básicos para criar programas, enquanto os *frameworks* fornecem um conjunto de diretrizes e funções para facilitar e acelerar o desenvolvimento de aplicações complexas (Farooq *et al.*, 2014). No Quadro 1, podem ser vistos a definição e o uso das linguagens e *frameworks* utilizados para o desenvolvimento do projeto.

Quadro 1 – Linguagens e *frameworks.*

Linguagens	Definição
Python	Python é uma linguagem de programação de alto nível, reconhecida pela clareza e simplicidade de seu código. Tal linguagem suporta múltiplos paradigmas de programação, destacando-se por sua legibilidade, o que permite aos desenvolvedores expressarem conceitos complexos de maneira concisa. (Van Rossum, 2018)
JavaScript	JavaScript (por vezes abreviado como JS) é uma linguagem de programação web utilizada por grande parte dos sites e navegadores. Ela é leve, interpretada e fundamentada em objetos, além de ser dotada de funções de primeira classe (Flanagan, 2020). Ademais, o TypeScript caracteriza-se como uma extensão do JavaScript, que adiciona recursos de tipagem estática opcional e outros aprimoramentos de desenvolvimento. (TYPESCRIPT, 2024)
HTML	HTML, ou *HyperText Markup Language* (Linguagem de Marcação de Hipertexto), é uma linguagem de marcação utilizada para estruturar o conteúdo de páginas da web. Desenvolvido para ser interpretado por navegadores, o HTML permite a criação de documentos que incluem texto, imagens, *links*, formulários e outros elementos interativos. (MDN WEB DOCS, 2024)
CSS	CSS, ou *Cascading Style Sheets* (Folhas de Estilo em Cascata), é uma linguagem de estilo utilizada em conjunto com o HTML para definir a apresentação e o layout de páginas web. Ela permite que os desenvolvedores controlem a aparência visual dos elementos HTML, aplicando estilos como cores, fontes, espaçamentos e posicionamentos. (MDN WEB DOCS, 2022)
C#	C# é uma linguagem de programação versátil, moderna e orientada a objetos, amplamente utilizada no desenvolvimento de aplicativos na plataforma .NET. Sua sintaxe clara, recursos avançados e integração com a plataforma .NET a torna uma escolha popular para uma variedade de cenários de desenvolvimento de software. (Microsoft Learn, 2024)
.NET	Dotnet (ou .NET) é uma plataforma de desenvolvimento que oferece um ambiente abrangente para criar uma variedade de aplicativos, de aplicações web e serviços a aplicativos para dispositivos móveis e sistemas corporativos. (Microsoft, 2023)
React Native	O React Native é um framework de desenvolvimento de aplicativos móveis que permite a criação de aplicações nativas para iOS e Android utilizando a linguagem de programação JavaScript e a biblioteca React. (React Native, 2024)
Serverless Framework	O *Serverless Framework* é um *framework* de arquitetura baseado na AWS que oferece uma abordagem para a criação de aplicativos em infraestrutura de nuvem, a qual pode ser escalonada automaticamente sem a incorrência de cobrança enquanto ociosa, e normalmente exige manutenção mínima. (Serverless Framework, 2024)

Fonte: elaboração dos autores (2025).

Lei Geral de Proteção de Dados (LGPD) e Protocolos de Segurança

A LGPD (Lei nº 13.709/2018) estabelece diretrizes para o tratamento de dados pessoais e impõe regulamentos rigorosos sobre a coleta, armazenamento e tratamento desses dados (Brasil, 2018).

Como o sistema interage com informações de compras, a conformidade com a LGPD mostrou-se essencial para garantir a privacidade dos dados dos usuários, sendo necessária a adoção das diretrizes da lei para assegurar a conformidade legal. (BNDES, 2021)

HTTP (*Hypertext Transfer Protocol*) é um protocolo fundamental utilizado na transferência de dados na *World Wide Web*. O *HTTPS* é uma extensão segura do *HTTP*, que utiliza o protocolo *SSL/TLS* (*Secure Sockets Layer/Transport Layer Security)* para criptografar os dados transferidos, garantindo que mesmo se forem interceptados, eles serão indecifráveis; além disso, o protocolo também verifica a identidade do servidor para o qual os dados estão sendo enviados, utilizando certificados digitais. (Rescorla, 2004)

Metodologia

Inicialmente, as investigações foram conduzidas tendo como base revisões bibliográficas de natureza exploratória, visando estabelecer uma base teórica que sustentasse o avanço do projeto. A primeira etapa consistiu em uma pesquisa teórica, seguida de uma revisão bibliográfica. Após isso, foi feito um *benchmark* para entender as tendências do mercado, buscando produtos semelhantes que já estejam em circulação. A seguir, a solução começou a ser construída, sendo criado, em primeiro lugar, o hardware, o aplicativo móvel e, por fim, a integração entre hardware e software. Em seguida, foram realizados os testes da solução em residências.

O projeto foi conduzido com o uso da metodologia ágil de desenvolvimento SCRUM, que consiste em um *framework* que organiza o desenvolvimento em ciclos de entregas incrementais e promove a divisão clara de responsabilidades entre os membros da equipe (SCRUM, 2024). Essa abordagem, juntamente com a elaboração de um *roadmap* para orientação das etapas do projeto, permitiu a construção progressiva de um produto final mais alinhado aos objetivos propostos.

Desenvolvimento do hardware

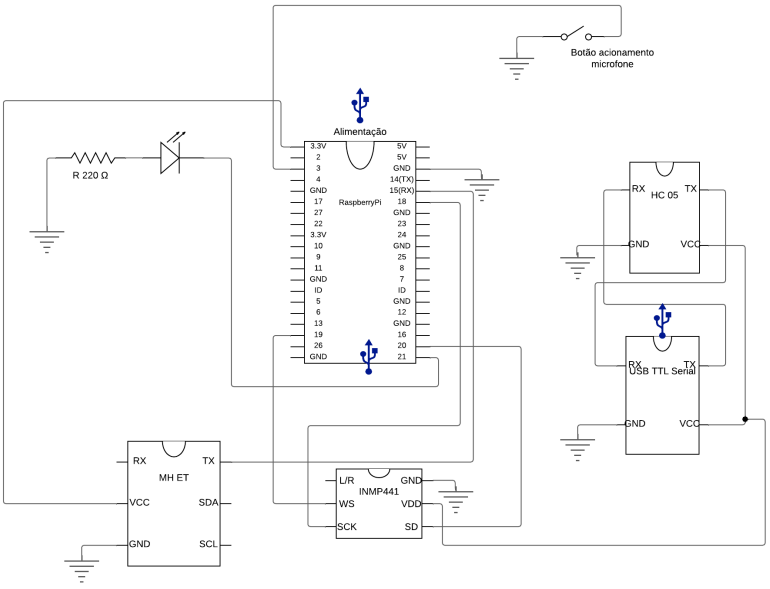
A concepção do hardware para este projeto baseou-se na integração de um conjunto de componentes essenciais, visando a criação da solução proposta. No centro do sistema, o Raspberry Pi 3B+ foi escolhido devido à sua capacidade computacional e ampla compatibilidade com periféricos. Para a captura de comandos de voz, o microfone INMP441 foi selecionado por sua alta sensibilidade e capacidade de registrar áudio de alta qualidade. A funcionalidade de leitura de códigos de barras foi incorporada através

do Scanner MH-ET Live, garantindo uma identificação rápida e precisa de produtos. Além disso, um *led*, um botão e um HC-05 foram integrados respectivamente como interfaces de usuário e conexão *bluetooth* facilitada.

Vale ressaltar que durante o desenvolvimento do projeto, surgiram desafios relacionados à escolha de hardware. Inicialmente, o ESP32-CAM foi testado, mas substituído devido à baixa qualidade das imagens e por ser um módulo descontinuado. Posteriormente, o ESP32 também foi descartado pela falta de memória e pela dificuldade em lidar com requisições HTTPS. O projeto, então, como supramencionado, foi construído utilizando o Raspberry Pi 3B+, que atendeu melhor às necessidades do sistema.

O desenvolvimento do protótipo iniciou-se com a montagem desses componentes conforme um esquema elétrico detalhado, que serviu como guia para a conexão dos elementos do sistema. Este esquema, apresentado na Figura 1, não apenas facilitou a montagem inicial, mas também forneceu uma base sólida para as etapas subsequentes do desenvolvimento.

Figura 1 – Esquema elétrico.



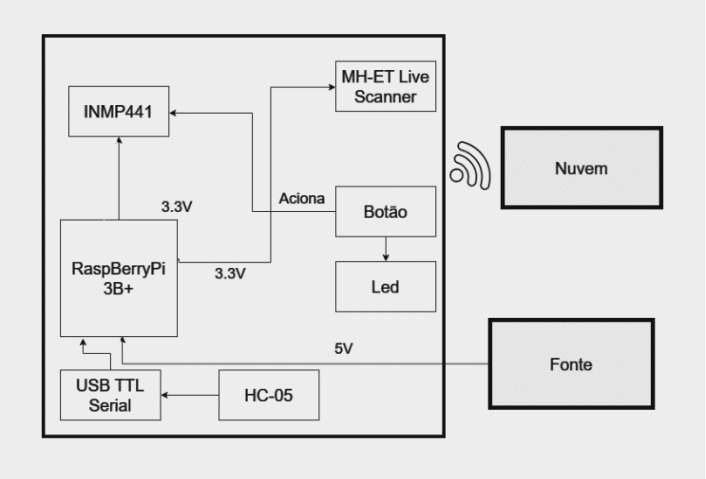
Fonte: elaboração dos autores. (2025)

Após a validação da usabilidade do protótipo na placa de prototipação pelo grupo, o projeto avançou para a transferência dos componentes para uma placa de fenolite. Esse passo foi essencial para a garantia da durabilidade do dispositivo, estabelecendo uma plataforma sólida para as etapas subsequentes de implementação do projeto, além de assegurar a sua finalidade em cenários práticos de uso. Por fim, para

proteger o hardware e melhorar sua estética, o conjunto completo foi alojado em uma caixa projetada especificamente para o dispositivo e produzida por uma impressora 3D. Este invólucro não só ofereceu proteção física aos componentes eletrônicos, mas também conferiu ao dispositivo um aspecto profissional e customizado.

Para uma compreensão detalhada da arquitetura e da disposição dos componentes de hardware utilizados neste projeto, a Figura 2 mostra um esquema ilustrativo da arquitetura do hardware.

Figura 2 – Diagrama de blocos da solução.



Fonte: elaboração dos autores. (2025)

Por fim, visando à experiência do usuário, foi pensado também na forma como a configuração seria feita pelo usuário final quando ele tivesse em mãos o dispositivo, de modo que ele não precisasse ligar o Raspberry Pi 3B+ e programar para se conectar ao wi-fi. Para solucionar tal problema foi definido na arquitetura do hardware um adaptador *bluetooth* via USB (HC-05) para que facilitasse a implementação e troca de informações entre a aplicação *mobile* e o hardware.

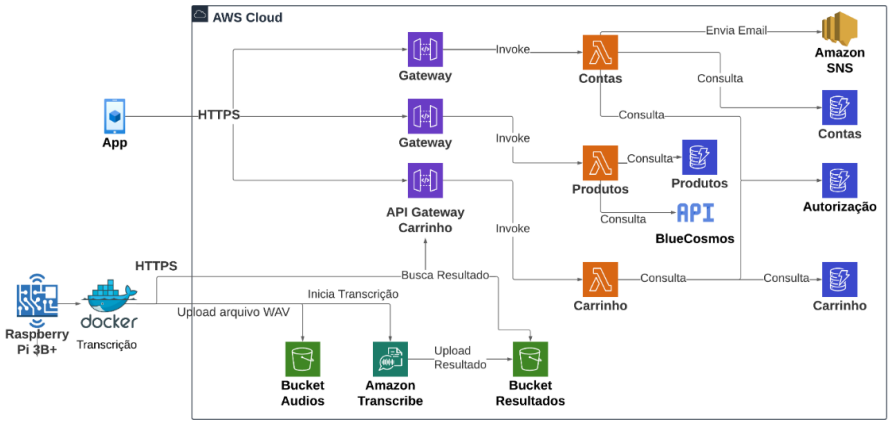
Desenvolvimento do aplicativo móvel

A solução envolveu a criação de um aplicativo móvel que capacitou os usuários a gerenciarem sua lista de compras de maneira simplificada e inteligente. Essa solução foi construída utilizando-se o *framework* de desenvolvimento React Native para desenvolver a interface *mobile*. Já o *back-end* da aplicação foi construído através de APIs *serverless* na linguagem C#, que, por sua vez, foram hospedados em nuvem no provedor AWS através do serviço *Lambda,* disponibilizado para web através do Amazon API *Gateway* e com o armazenamento dos dados persistentes através do Amazon DynamoDB.

A API de contas utilizou um sistema feito em C# para gerar senhas de uso único, permitindo que os usuários realizassem login, e gerassem tokens únicos para sua identificação nas outras APIs, persistindo os dados em uma tabela do DynamoDB. Já a API de produtos utiliza um sistema em C# para consultar os dados de produtos (nome, código de barras e URL da imagem) e retorná-los ao cliente. A API de carrinho armazena os produtos no carrinho de cada usuário através da criação de objetos no DynamoDB, utilizando também um sistema em C#. A API de transcrição recebe um arquivo WAV, transcreve o texto para obter o produto e a quantidade e aciona a API de carrinho para adicionar o produto, operando com um sistema em C#.

Na Figura 3, pode-se acompanhar o funcionamento do sistema na nuvem de forma visual, com cinco APIs, sendo três delas acessadas pelo App e duas acessadas pelo dispositivo físico, sendo elas: API de contas, API de produtos, API de carrinho, API de transcrição e API da BlueCosmos, para quando não for encontrado o produto no banco de dados.

Figura 3 - Arquitetura geral.



Fonte: elaboração dos autores. (2025)

Ademais, vale ressaltar que se fez necessária a compra de um banco de dados de produtos, visando à rapidez da aplicação para o usuário final na leitura dos códigos de barras e reconhecimento do produto. Porém, mesmo com um banco de dados amplo, foi implementada uma integração com a API BlueCosmos para os casos em que o produto não se encontra na base de dados. Entretanto, apesar de poder ser utilizada para essa alternativa, é inviável tentar usá-la individualmente devido ao limite de 25 consultas diárias no nível gratuito. A API BlueCosmos pode ser acessada através do *link* https://cosmos.bluesoft.com.br (Cosmos, 2024).

Além disso, as senhas foram armazenadas em formato de *hash*, através do algoritmo SHA256, impossibilitando o acesso a dados confidenciais. O algoritmo foi escolhido devido ao equilíbrio entre segurança e desempenho, garantindo a segurança sem comprometer o desempenho (Nist, 2012).

Jornada do usuário

A jornada do usuário foi essencial para entender como o indivíduo interage com o sistema, desde a configuração inicial do dispositivo até o uso diário. A análise detalhada de cada interação do usuário, ilustrada na Figura 4, permitiu otimizar as interfaces e melhorar continuamente a interação com o aplicativo.

Figura 4 – Mapa da jornada do usuário.

Fonte: elaboração dos autores (2025).

Além do mais, para que o usuário conseguisse entender como configurar o seu dispositivo pela primeira vez e como utilizá-lo após essa configuração foram criados vídeos que estão disponíveis no Youtube, no canal oficial do CartNinja: https://www.youtube.com/@CartNinjaOficial.

Integração entre hardware e aplicativo móvel

O processo de implementação englobou a interconexão entre o hardware e as APIs através de uma arquitetura *serverless* que constituíram a infraestrutura da solução. A implementação *API gateways* na

arquitetura do projeto permite que as funções no AWS *Lambda* sejam acessadas através da internet, fazendo com que as partes do sistema se comuniquem através do protocolo *HTTPS*.

A leitura do código de barras é feita pelo scanner MH-ET Live, que envia os dados para a API de carrinho através do próprio Raspberry Pi 3B+, com um código em python. Em seguida, a aplicação acessa o banco de dados com os produtos já inseridos.

O recurso de *speech to text* foi implementado de forma que, ao clicar no botão, o microfone capta o áudio por 4 segundos, transmitindo os dados para Raspberry Pi 3B+ em formato WAV, que, mesmo sendo um arquivo relativamente pesado quando comparado com outros formatos, foi mantido devido ao fato de não ser comprimido, mantendo o áudio do jeito que é recebido e sem perda de qualidade (IBM; Microsoft, 1991). Em seguida, os dados são enviados para uma API rodando em um contêiner Docker no próprio Raspberry Pi 3B+. Esta API envia o arquivo WAV para uma API de transcrição em nuvem, que realiza a conversão de fala para texto, o qual, uma vez transcrito, é então capturado e enviado para a API de carrinho, que insere o produto na lista de compras.

Trabalhando em conjunto com o restante do hardware, a solução permite que os usuários tenham melhor controle dos produtos em sua dispensa ao automatizar o processo de criação de uma lista de compras virtual. Dessa maneira, retirou-se a etapa de escolher manualmente produtos domésticos que acabaram, evitando duplicação na hora de realizar as compras.

Teste da solução

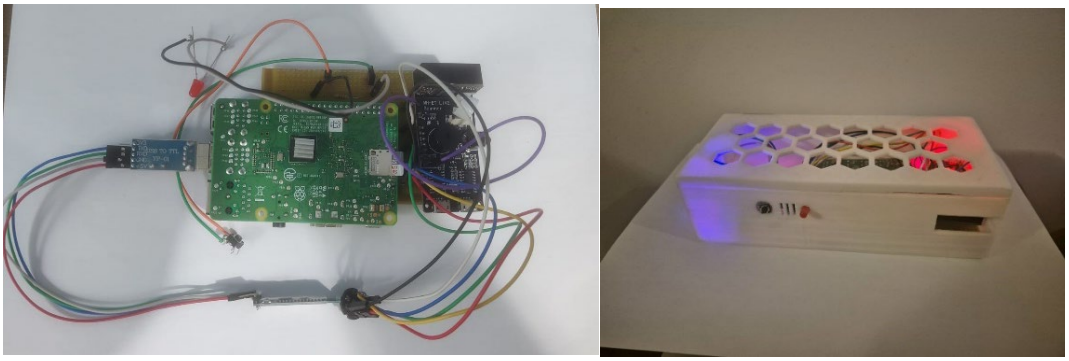
Para validar a solução, foram realizadas pesquisas com os usuários envolvidos. A primeira, antes da implementação da solução, com o objetivo de detectar a situação atual de desperdício de alimentos na residência do usuário. A segunda pesquisa teve o objetivo de realizar uma avaliação *in loco* da percepção do usuário em relação ao funcionamento do sistema. Após isso, os retornos dos usuários foram analisados e as melhorias ao longo dos testes foram implementadas.

Todos os testes foram realizados com o consentimento dos usuários no momento da coleta de dados, sendo disponibilizados termos de uso claros e transparentes, informando de maneira detalhada como as informações seriam utilizadas, armazenadas e protegidas, de forma a seguir os princípios vigentes da LGPD. Os formulários utilizados para a coleta de informações foram aplicados em dois momentos: antes do uso da aplicação, por meio do formulário disponível no *link* https://repocartninja.s3.sa-east-1.amazonaws.com/Formulario-pre-teste.pdf, e após a utilização da aplicação, por meio do formulário disponível no *link* https://repocartninja.s3.sa-east-1.amazonaws.com/Formulario-pos-teste.pdf.

Resultados e discussão

No desenvolvimento deste projeto, a interação entre a concepção teórica e a prática foi essencial para superar os desafios e obter uma solução viável e funcional. Na Figura 5, à esquerda, é possível observar o protótipo colocado na placa de fenolite, e, à direita, a versão final do protótipo.

Figura 5 – Protótipo na placa de fenolite (à esquerda) e versão final do protótipo (à direita).



Fonte: elaboração dos autores (2025).

Essas figuras mostram que este protótipo incorporou não apenas a funcionalidade robusta e a interatividade do sistema, mas também refletiu a atenção dedicada à estética e ao design prático.

Viabilidade financeira

Este projeto também contemplou uma avaliação criteriosa da viabilidade financeira, focalizando os custos associados ao desenvolvimento do protótipo. Essa análise incluiu, principalmente, a aquisição dos componentes. A transparência na divulgação dos custos foi crucial para assegurar a replicabilidade e a sustentabilidade do projeto, permitindo a compreensão clara dos recursos financeiros necessários para sua execução.

Para o protótipo, foram gastos R$ 816,27 abrangendo a aquisição de componentes essenciais. Entre os itens, destacam-se o Raspberry Pi 3B+ (R$ 380,00), o scanner MH-ET Live (R$ 300,00), o adaptador TTL Serial USB (R$ 18,27), o módulo HC 05 (R$ 35,00), o microfone INMP441 (R$ 38,00), a placa de fenolite (R$ 20,00) e a resina para fabricação da caixa 3D (R$ 25,00). Além disso, foram investidos R$ 120,00 em um banco de dados de produtos, garantindo a infraestrutura básica.

Os custos mensais para a manutenção da infraestrutura em nuvem, previstos para 100 usuários, totalizam R$13,13. Esses custos incluem AWS *Lambda* ($ 0,15), Amazon DynamoDB ($ 0,51), Amazon API *Gateway*

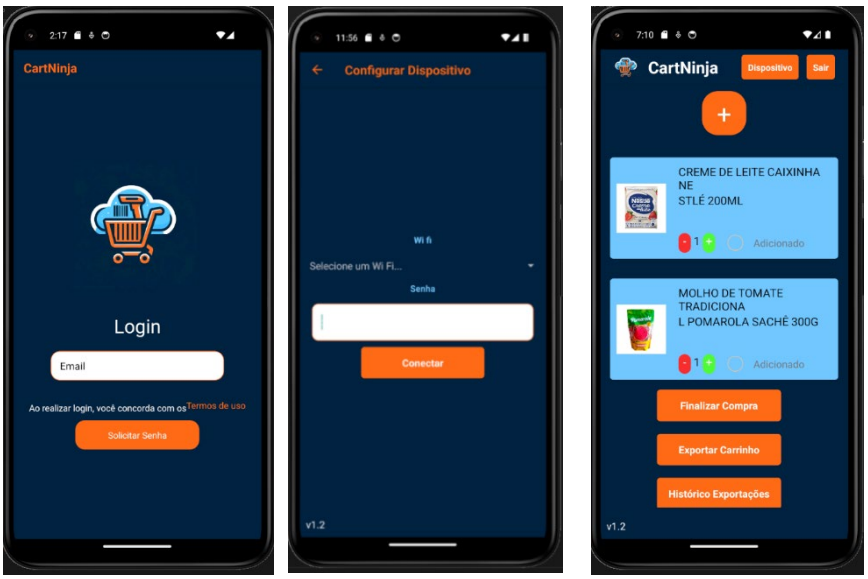
($ 0,35), Amazon *Transcribe* ($ 12,00) e Amazon S3 ($ 0,12). Essa estrutura garante o funcionamento contínuo do sistema e sustenta a escalabilidade do projeto. Todo esse cálculo pode ser acessado através do *link* da calculadora de AWS:

https://calculator.aws/#/estimate?id=ab35ecab3fb793fca6e2c46c5e644c368e3bb51c.

Aplicativo móvel

O aplicativo móvel foi desenvolvido para facilitar a experiência do usuário, adotando um design minimalista e intuitivo. A interface de login é simples e inclui o *link* para os termos de uso, que podem ser conferidos através do *link* https://repocartninja.s3.sa-east-1.amazonaws.com/Termos+de+Uso+TCC+Carrinho+de+Compras+Virtual.pdf, garantindo, assim, conformidade com a LGPD. As telas do aplicativo móvel podem ser vistas na Figura 6, da esquerda para a direita, respectivamente, contemplando a interface de login, a interface de configuração do dispositivo e a interface principal, todas elas com um design limpo e intuitivo.

Figura 6 – Abas do aplicativo móvel.



Fonte: elaboração dos autores (2025).

A tela de configuração do dispositivo mantém a consistência estética com a tela de login. Ela foi criada com o intuito de informar ao usuário acerca do estado atual da conexão do dispositivo. O design também foi criado para ser direto, focando na funcionalidade essencial da configuração do dispositivo. Já a interface principal do aplicativo ficou responsável por ser a área onde os usuários gerenciam sua lista de compras virtual. Os produtos foram listados com opções para ajustar quantidades a um indicador visual de 'Adicionado', realçando o status do produto no carrinho. Botões de ação '+' para adição de produtos e 'Finalizar Compra' são elementos proeminentes, promovendo uma experiência de usuário fluida e eficaz.

Ademais, nessa aba também tem a opção de 'Exportar Carrinho' e verificar o histórico de exportações, o que permite que a lista de compras seja enviada e compartilhada entre as pessoas.

Esse design do aplicativo móvel refletiu os objetivos do projeto de simplificar a montagem de uma lista de compras. Para tal função, o uso de cores contrastantes utilizando a paleta de cores da identidade visual do produto não apenas melhorou a usabilidade, mas também criou uma experiência visual agradável. O aplicativo pode ser baixado pelo *link:* https://repocartninja.s3.sa-east-1.amazonaws.com/app-release.apk.

Resultados da usabilidade da solução

Para testar a solução, foram construídos dois dispositivos, que foram testados em 3 (três) residências diferentes por um período de três semanas, para avaliar e verificar se o objetivo do projeto foi atingido, além de obter retorno dos usuários finais.

Na primeira residência, apesar do aplicativo estar em uma versão inicial sem identidade visual, os usuários elogiaram sua interface intuitiva e a conveniência de adicionar produtos por comandos de voz. Como melhoria, sugeriram a funcionalidade de exportar carrinho, que foi implementada nas residências seguintes. Já na segunda residência, destacou-se a eficiência do scanner de códigos de barras e a integração com a rotina diária. Contudo, foram apontadas falhas no reconhecimento de voz em ambientes ruidosos e lentidão na entrada de produtos por comando de voz. Por fim, na terceira residência, o sistema foi elogiado por incentivar o planejamento de compras, mas houve sugestão para uma interface mais personalizável, com abas para uso individual por membros da família.

Análise comparativa das residências

Antes da implementação da aplicação, os usuários relataram dificuldades no gerenciamento do estoque doméstico, com frequentes idas ao supermercado devido a itens esquecidos ou compras duplicadas, gerando estoque desnecessário. Os pré-formulários também indicaram interesse em uma solução que ajudasse na organização das compras e redução de idas ao mercado.

Após a utilização da aplicação, os resultados foram positivos, sendo que nas três residências foram destacadas a facilidade de se criar listas de compras, melhorando a gestão de estoque e tornando as compras mais eficientes. Apesar de não haver redução no número de idas ao supermercado devido ao curto período de testes, muitos relataram satisfação geral e indicaram que recomendariam a tecnologia. Problemas menores, como falhas no reconhecimento de voz em ambientes ruidosos, demora na adição de itens ao carrinho por voz e ausência de integração com carrinhos para exportação foram mencionados,

mas não comprometeram a experiência. Além disso, a economia de dinheiro foi frequentemente apontada como benefício, graças à reposição mais eficiente do estoque residencial.

Os usuários sugeriram diversas melhorias para a aplicação, como a personalização de notificações, criação de abas para uso individual por membros da família e funcionalidades para direcionar compras a mercados específicos, eliminando a necessidade de sair de casa. Apesar das experiências dos usuários terem sido amplamente positivas, essas sugestões apontam áreas importantes para futuras atualizações e aperfeiçoamentos.

Código-fonte

No Quadro 3, estão presentes os *links* dos repositórios do GitHub que foram utilizados para o versionamento do código.

Quadro 2 – Códigos e repositórios

Descrição	***Link***
Código da aplicação *mobile* utilizando React Native	https://github.com/ProjetoRepositor/app-mobile
Códigos em python e *shell script* que configuram o RaspBerry Pi 3B+ para fazer a leitura de código de barras, leitura de áudio e configuração do *wi-fi* por *bluetooth*.	https://github.com/ProjetoRepositor/firmware
API de carrinho para adicionar produtos na lista de compras seguindo o padrão *serverless* em nuvem.	https://github.com/ProjetoRepositor/backend-carrinho
API de transcrição que roda dentro de um container Docker no RaspBerry Pi 3B+.	https://github.com/ProjetoRepositor/backend-api-transcricao
API de produtos que faz a busca pelo código de barras, seguindo o padrão *serverless* em nuvem.	https://github.com/ProjetoRepositor/backend-api-produtos
Arquivo em *shell script* para configurar automaticamente o raspberry na primeira vez em que for ligado.	https://github.com/ProjetoRepositor/setup-raspberry
API de contas que configura o acesso dos usuários seguindo o padrão *serverless* em nuvem.	https://github.com/ProjetoRepositor/backend-contas

Fonte: elaboração dos autores (2025)

Considerações finais

Este artigo desenvolveu uma solução para auxiliar na redução do desperdício alimentar residencial a partir da simplificação do processo de montagem de uma lista de compras, utilizando leitura de código de barras e reconhecimento de voz. O projeto teve em sua composição um aplicativo móvel construído com *React Native*, uma infraestrutura *serverless* em nuvem para suportar as APIs e um banco de dados em nuvem,

além de componentes de hardware que compuseram a solução *IoT*, a fim de tornar possível a adição de produtos à lista do usuário de forma segura, escalável e resiliente.

Por limitações de tempo e recursos, os testes foram realizados em apenas 3 residências por um curto período, sendo possível efetuar apenas uma análise qualitativa desses resultados obtidos. Recomenda-se, então, para projetos futuros a ampliação da quantidade e duração dos testes, o que permitirá a realização de análises quantitativas sobre a durabilidade do hardware, escalabilidade e impacto da solução na redução do desperdício alimentar. Além disso, vale ressaltar que a ideia original do projeto também contava com o encaminhamento direto para aplicativos ou mercados para reabastecimento automático de estoque. Essa funcionalidade pode ser desenvolvida como uma melhoria futura, permitindo entregas automáticas e eliminando a necessidade de deslocamento do usuário.

Para melhorias futuras, sugere-se ainda substituir a API da AWS por um modelo de *machine learning* com treinamento local, acelerando a transcrição de voz, tirando, assim, a dependência de uma API externa de transcrição e contribuindo para acelerar esse processo de adição de itens no carrinho quando feito por voz. Ademais, o projeto enfrenta desafios associados à transição deste protótipo para um produto com produção em larga escala, sendo um dos principais obstáculos a necessidade de aprovação por órgãos regulatórios, como a Agência Nacional de Telecomunicações (ANATEL) no Brasil. Por fim, embora o protótipo tenha demonstrado potencial, a escalabilidade do projeto exige uma análise cuidadosa dos aspectos de fabricação e distribuição, sendo que mudanças significativas nos componentes de hardware podem ser necessárias para atender requisitos de custo, durabilidade, sustentabilidade e eficiência em uma operação de escala comercial.

Por fim, um relatório técnico detalhado foi elaborado para apresentar o projeto, de forma a tentar vender a solução desenvolvida, e está disponível no *link* https://www.canva.com/design/DAGC47Yilfg/8v0NB1bcMFobiW3HtXoAug/edit?utm_content=DAGC47Yilfg&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton.

Referências

AWS. Amazon web documentation, 2024. AWS, 2024. Disponível em: https://docs.aws.amazon.com/. Acesso em: 6 maio 2024.

BASNIAK, M.I.; LIZIERO, A.R.. A impressora 3D e novas perspectivas para o ensino: possibilidades permeadas pelo uso de materiais concretos. **Revista Observatório**. [S. l.], v. 3, n. 4, p. 445–466, 2017. DOI: 10.20873/uft.2447-4266.2017v3n4p445. Disponível em:

https://sistemas.uft.edu.br/periodicos/index.php/observatorio/article/view/3321. Acesso em: 14 jun. 2024.
BLUESOFT Cosmos. **O que é o bluesoft cosmos?**. Disponível em: https://cosmos.bluesoft.com.br/o-que-e-o-cosmos. Acesso em: 10 set. 2023.
BRASIL. Lei Nº 13.709, de 14 de agosto de 2018. Lei Geral de Proteção de Dados Pessoais (LGPD). Diário Oficial da União, Brasília, DF, 15 ago. 2018. Disponível em: https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm. Acesso em: 30 de novembro de 2023.
BRASIL. Lei Geral de Proteção de Dados (LGPD). BNDES, 2024. Disponível em: https://www.bndes.gov.br/wps/portal/site/home/transparencia/lgpd. Acesso em: 12 maio 2024.
DANZIGER, Pamela N. Smart carts may be the disruptive technology grocery stores need now. Forbes, 3 ago. Disponível em: https://www.forbes.com/sites/pamdanziger/2023/08/03/smart-carts-maybe-the-disruptive-technology-grocery-stores-need-now/?sh=42444ab64588. Acesso em: 6 maio 2024.
DOCKER. **Docker Docs**. Disponível em: https://docs.docker.com/guides. Acesso em: 22 maio 2024.
FAROOQ, M. S. *et al*. An evaluation framework and comparative analysis of the widely used first programming languages. **PLoS ONE**. South Africa, v. 9, n. 2, p. e88941, 24 fev. 2014. Disponível em: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0088941. Acesso em: 20 maio 2024.
FARR-WHARTON, G.; FOTH, M. V; CHOI, J. H.-J. Identifying factors that promote consumer behaviours causing expired domestic food waste: Factors promoting behaviours causing food waste. **Journal of consumer behaviour**. USA, v. 13, n. 6, p. 393–402, 2014. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1002/cb.1488. Acesso em: 27 ago. 2023.
FLANAGAN, D. **JavaScript**: Master the World's Most-Used Programming Language. Sebastopol: O'Reilly Media, Incorporated, 2020.
GOOGLE cloud. **O que é computação na nuvem?.** Disponível em: https://cloud.google.com/learn/what-is-cloud-computing?hl=pt-br#:~:text=O%20Google%20Cloud%20. Acesso em: 12 maio 2024.
IBM. **O que é uma API de REST?.** Disponível em: https://www.ibm.com/br-pt/topics/rest-apis. Acesso em: 12 maio 2024.
INSTITUTO Brasileiro de Geografia e Estatística (IBGE). **Pesquisa de orçamentos familiares (POF) 2017-2018**: Análise da Segurança Alimentar no Brasil. Rio de Janeiro: IBGE, 2019. Disponível em: https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/28903-10-3-milhoes-de-pessoas-moram-em-domicilios-com-inseguranca-alimentar-grave. Acesso em: 26 ago. 2023.
MDN. **MDN web docs**. Disponível em: https://developer.mozilla.org/pt-BR/. Acesso em: 6 maio 2024.
MICROSOFT. **.NET**. Disponível em: https://dotnet.microsoft.com/pt-br/. Acesso em: 12 maio 2024.
MICROSOFT LEARN. **C#.** https://learn.microsoft.com/pt-br/dotnet/csharp/. Acesso em: 13 maio 2024.
NIST. **Recommendation for applications using approved hash algorithms**. Disponível em: https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-107r1.pdf. Acesso em: 12 junho 2024.
ONU. **Goal 2**: Zero hunger. Disponível em: https://www.un.org/sustainabledevelopment/hunger/. Acesso em: 29 ago. 2023.
PRAJWAL, T.; SOWMYA, K. B. **Inter-IC sound (I2S) interface for dual mode bluetooth controller**. Singapore: Springer, 2024.Disponível em: https://link.springer.com/chapter/10.1007/978-981-16-8763-1_32. Acesso em: 13 mar. 2024.
PROGRAMA DAS NAÇÕES UNIDAS PARA O MEIO AMBIENTE (PNUMA). **Food waste index report 2021**. Nairobi: UNEP, 2021. Disponível em: https://www.unep.org/pt-br/resources/relatorios/indice-de-desperdicio-de-alimentos-2021#:~:text=As%20estimativas%20sugerem%20que%208,pessoas%20e%20para%20o%20planeta. Acesso em: 22 ago. 2023.
RASPBERRY Pi Foundation. **Raspberry pi foundation**: About Us. Disponível em: https://www.raspberrypi.org/about/. Acesso em: 6 maio 2024.

REACT Native. **Core components and APIs.** Disponível em: https://reactnative.dev/docs/components-and-apis. Acesso em: 6 maio 2024.
REIS, S. N.; HIUGA, S. H..; Lupinetti, V.A.; Oliveira, W.A. **Monitoramento de despensa residencial com auxílio da assistente virtual Alexa**. Biblioteca Virtual CEFSA. Disponível em: https://cefsa.bnweb.org/bnportal/em/pt-BR/search?exp=Monitoramento%20de%20despensa%20residencial%20com%20aux%C3%ADlio%20da%20assistente%20virtual%20Alexa%20. Acesso em: 10 ago. 2023.
RESCORLA, E**. SSL and TLS**: designing and building secure systems. Boston: Addison-Wesley, 2004. Disponível em: https://dl.acm.org/doi/10.5555/358346. Acesso em: 12 maio 2024.
SCRUM. **What is Scrum?**. Disponível em: https://www.scrum.org/learning-series/what-is-scrum/what-is-scrum. Acesso em: 22 junho 2025.
SERVERLESS. **The serverless application framework**. Disponível em: https://www.serverless.com/. Acesso em: 6 maio 2024.
SIAW-YEBOAH, F. et al. GHBS-13 as an enhanced EAN-13 barcode module for monitoring processed consumable products in Ghana. Engineering reports. [S. l.], v. 5, n. 6, 20 dez. 2022. Disponível em: https://onlinelibrary.wiley.com/doi/10.1002/eng2.12612?af=R. Acesso em: 10 fev. 2024.
TYPESCRIPT. **Documentation typeScript for the new programmer**. Disponível em: https://www.typescriptlang.org/docs/handbook/typescript-from-scratch.html. Acesso em: 12 maio 2024.
VAN ROSSUM, G. (Python development team). **The python library reference**. [S.l: s.n.], 2018. Disponível em: https://dl.acm.org/doi/book/10.5555/3217659. Acesso em: 15 jan. 2024.
VON PFLUG, P.; KRISCHKER, D. **Aspects of the use of MEMS microphones in phased array systems.** Proceedings of Internoise, Germany, v. 13, n. 6, p. 393–402, 2014. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1002/cb.1488. Acesso em: 27 ago. 2023.