

Sistema de Apoio à Decisão Nutricional com Algoritmo KNN

Roger da Palma Culau, Herysson Rodrigues Figueiredo

Curso de Ciência da Computação

UFN - Universidade Franciscana

Santa Maria - RS

Email: roger.palma@ufn.edu.br, herysson.figueiredo@ufn.edu.br

Resumo—Este trabalho apresenta um Sistema de Apoio à Decisão voltado à nutrição personalizada, com o objetivo de ajudar nutricionistas na recomendação de alimentos nutricionalmente semelhantes. A Tabela Brasileira de Composição de Alimentos (TACO) foi utilizada como base de dados. O algoritmo *K-Nearest Neighbors* (KNN) foi aplicado para calcular a similaridade com base em proteínas, carboidratos, lipídios, fibras, valor energético, entre outros. A metodologia seguiu o framework *Scrum*, adaptado ao contexto acadêmico. O sistema foi implementado em *Python*, *React* e *SQLite*. A interface permite a entrada de dados e definição de prioridades. Após validação, o sistema contribui com recomendações de alimentos de forma ágil e personalizada.

Palavras-chave: Nutrição Personalizada, *K-Nearest Neighbors*, Aprendizado de Máquina, tabela TACO.

I. INTRODUÇÃO

A nutrição desempenha um papel fundamental na manutenção da saúde, prevenindo doenças crônicas e promovendo qualidade de vida. No entanto, a escolha de refeições adequadas para indivíduos com restrições alimentares ainda representa um desafio. Estudos apontam que a alimentação inadequada é um dos principais fatores de risco para doenças como diabetes e hipertensão [1].

A integração de Sistemas de Apoio à Decisão (SAD) no domínio da nutrição tem se mostrado promissor para aprimorar a personalização das recomendações alimentares, oferecendo recomendações personalizadas. Sistemas de Apoio à Decisão, quando combinados com Tomada de Decisão Multicritério ou (*Multi-Criteria Decision Making* - MCDM) ¹ têm sido amplamente aplicados em diversas áreas, incluindo saúde, agricultura e gestão de recursos, tornando-se uma ferramenta fundamental para a tomada de decisões complexas [2], [4].

As dietas restritivas, quando não acompanhadas por um profissional, podem levar a sérios problemas de saúde. Muitas pessoas fazem mudanças na alimentação sem orientação adequada, o que pode resultar na falta de nutrientes essenciais para o funcionamento do organismo. Segundo Sonia Trecco, nutricionista do Hospital das Clínicas (HC) da Faculdade de Medicina da Universidade de São Paulo (FMUSP), "o nosso organismo necessita de todos os nutrientes, alguns em maior e outros em menor quantidade. Na falta deles, podemos ter

carências nutricionais" [1]. Além disso, a restrição alimentar excessiva pode gerar um efeito reverso, levando ao descontrole e a episódios de compulsão alimentar. Estudos indicam que 95% das pessoas que fazem dietas restritivas voltam a ganhar peso posteriormente, pois o cérebro interpreta a restrição como um estado de privação e busca recuperar as reservas de gordura assim que a alimentação normal é retomada [1].

Para garantir a precisão e a confiabilidade das informações nutricionais utilizadas nas recomendações, o sistema utilizará como base principal a Tabela Brasileira de Composição de Alimentos – TACO, 4ª edição revisada e ampliada. Essa tabela foi elaborada pelo Núcleo de Estudos e Pesquisas em Alimentação (NEPA) da Universidade Estadual de Campinas (UNICAMP), sendo considerada uma das principais fontes nacionais de dados nutricionais detalhados sobre alimentos consumidos no Brasil [6].

A. Objetivo Geral

Desenvolver um Sistema de Apoio à Decisão (SAD) para auxiliar nutricionistas na substituição e recomendação de alimentos com base em perfis nutricionais semelhantes.

B. Objetivos Específicos

- Identificar alimentos com perfil nutricional semelhante, considerando variáveis como valor calórico, macronutrientes e micronutrientes;
- Alimentar um banco de dados com as informações da Tabela Brasileira de Composição de Alimentos (TACO) como fonte principal de dados nutricionais;
- Implementar o algoritmo *KNN K-Nearest Neighbors*, utilizando *Python*, para calcular a similaridade entre alimentos;
- Oferecer recomendações de substituições alimentares ao nutricionista, de forma personalizada conforme as necessidades nutricionais do paciente;
- Desenvolver o sistema web utilizando tecnologias como *HTML*, *CSS*, *React*, *React Native* e *Expo*.
- Validar o sistema por meio de testes com diferentes cenários simulados, avaliando usabilidade, precisão das recomendações e desempenho técnico.

¹Métodos usados para considerar múltiplos critérios simultaneamente na tomada de decisão, comuns em problemas complexos como seleção de alternativas em saúde ou logística.

II. REFERENCIAL BIBLIOGRÁFICO

O referencial reúne informações sobre fundamentos de nutrição, tabelas de composição de alimentos, Sistemas de Apoio a Decisão e Algoritmos *KNN* e Ferramentas e tecnologias.

A. Fundamentos da Nutrição e Dietas com Restrições

A nutrição é a ciência que estuda os processos pelos quais o organismo ingere, digere, absorve, transporta, utiliza e excreta os nutrientes. Esses processos são essenciais para a manutenção da vida, o crescimento, a reparação tecidual e a prevenção de doenças. Os princípios básicos da nutrição envolvem a ingestão equilibrada de macronutrientes (carboidratos, proteínas e lipídeos) e micronutrientes (vitaminas e minerais), adequados às necessidades fisiológicas e às condições clínicas individuais [10].

No contexto clínico, a terapia nutricional requer a consideração de diretrizes alimentares baseadas em evidências, respeitando aspectos culturais, estado clínico e necessidades nutricionais específicas de cada paciente. O processo de cuidado nutricional inclui etapas como avaliação, diagnóstico, intervenção e monitoramento, sendo um ciclo contínuo que visa promover o estado nutricional adequado [11].

As dietas com substituições alimentares são ferramentas fundamentais no tratamento e prevenção de agravos à saúde. Elas podem ser definidas com base em nutrientes como as dietas hipossódicas (com baixo teor de sódio, indicadas para controle da pressão arterial) ou hipoglicídicas (com baixo teor de carboidratos, utilizadas no controle glicêmico), em intolerâncias alimentares, como as dietas isentas de lactose ou glúten, ou ainda em condições clínicas específicas. Tais dietas podem ser temporárias ou permanentes, a depender da patologia ou do objetivo terapêutico [12].

Em casos de distúrbios neurológicos com comprometimento da deglutição, como disfagia², são necessárias modificações de textura dos alimentos (graus pastosos, mecanicamente alterado, macio), além da consistência dos líquidos, garantindo segurança alimentar e aporte nutricional adequado. A escolha da textura depende do grau da disfagia e da capacidade funcional do paciente [10].

Para condições crônicas, como hipertensão ou doenças cardiovasculares, o planejamento alimentar pode incluir dietas como a abordagens dietéticas para parar a hipertensão (*Dietary Approaches to Stop Hypertension - DASH*)³, com restrição de sódio, incremento de potássio, cálcio e magnésio, e ênfase no consumo de frutas, verduras e alimentos integrais. A adaptação dietética visa controlar fatores de risco e prevenir complicações clínicas [10].

²Disfagia é a dificuldade de engolir alimentos, líquidos ou saliva, podendo comprometer a ingestão adequada de nutrientes. Está frequentemente associada a condições neurológicas, envelhecimento ou doenças do trato digestivo, exigindo adaptações na dieta. [10]

³DASH é um plano alimentar desenvolvido para prevenir e controlar a hipertensão, com ênfase em frutas, vegetais, grãos integrais, laticínios com baixo teor de gordura e redução do consumo de sódio, gorduras saturadas e açúcares.

B. Tabela Brasileira de Composição de Alimentos (TACO)

A compreensão dos fundamentos da nutrição e das necessidades específicas de cada paciente exige acesso a dados confiáveis sobre a composição dos alimentos. Para apoiar essa análise nutricional precisa, recorre-se a bases estruturadas que detalham os macro e micronutrientes presentes nos alimentos.

A Tabela Brasileira de Composição de Alimentos (TACO), desenvolvida pelo Núcleo de Estudos e Pesquisas em Alimentação (NEPA) da UNICAMP, é uma das principais referências nacionais para composição centesimal⁴, de minerais, vitaminas, aminoácidos e valor energético de alimentos consumidos no Brasil [6]. Trata-se de uma ferramenta essencial para análises nutricionais, formulação de dietas, rotulagem de produtos.

Lançada em sua primeira versão em 2004, a TACO passou por atualizações que ampliaram significativamente o número de alimentos e nutrientes analisados. A 4ª edição de 2011 revisada e ampliada contempla informações sobre 597 alimentos, com análises conduzidas por laboratórios credenciados e utilizando metodologias rigorosas de amostragem e validação analítica, o que garante confiabilidade aos dados disponibilizados.

Tabela I
COMPOSIÇÃO NUTRICIONAL DE CARNES SELECIONADAS

Cód.	Alimento	Kcal	Prot. (g)	Lip. (g)	Carb. (g)
323	Apresentado	129	13,5	6,7	2,9
326	Acém moído cozido	212	26,7	10,9	0,0
327	Acém moído cru	137	19,4	5,9	0,0
331	Almôndega frita	272	18,2	15,8	14,3
334	Contra-filé c/ gordura cru	217	19,2	15,0	0,0
337	Contra-filé s/ gordura grelh.	239	35,1	10,0	0,0
338	Charque cozido	263	36,4	11,9	0,0
340	Bovina à milanesa	352	20,6	24,0	12,2

A tabela I apresenta um exemplo da composição nutricional de algumas carnes, conforme os dados extraídos da TACO. Além desses dados principais, também são apresentados micronutrientes como cálcio, ferro, fósforo, potássio, bem como vitaminas do complexo B (tiamina, riboflavina, niacina) e vitamina C.

C. Sistemas de Apoio à Decisão

A complexidade envolvida na seleção de alternativas alimentares compatíveis com objetivos específicos exige mais do que simples consulta a dados. É nesse cenário que os Sistemas de Apoio à Decisão (*SAD*) ganham relevância, ao proporcionar ferramentas capazes de processar grandes volumes de informação e gerar recomendações alinhadas a múltiplos critérios nutricionais.

Os *SAD* constituem uma classe de sistemas computacionais interativos utilizados para auxiliar gestores e profissionais na resolução de problemas por meio da integração entre dados, modelos e técnicas analíticas.

Esses sistemas foram desenvolvidos para fornecer suporte à tomada de decisão, e não para substituí-la, respeitando a

⁴Composição centesimal refere-se à quantificação dos nutrientes presentes em 100 gramas do alimento, permitindo a padronização e a comparação entre diferentes itens alimentares.

complexidade dos contextos organizacionais e a subjetividade inerente aos julgamentos humanos [2].

Um sistema de apoio a decisão é composto essencialmente por três componentes principais: (i) uma base de dados com informações relevantes ao problema, (ii) uma base de modelos com métodos quantitativos e analíticos (como simulações, árvores de decisão, heurísticas e algoritmos), e (iii) uma interface amigável que permita a interação do decisor com o sistema [2].

O SAD foca na análise preditiva e na simulação de cenários futuros, permitindo responder a perguntas do tipo “What-if?”, que consistem em testar diferentes combinações de variáveis para prever possíveis resultados, sendo fundamentais para decisões com múltiplos critérios e objetivos. A flexibilidade é uma de suas principais características, tornando-o útil em ambientes dinâmicos e incertos. [13].

Além disso, a evolução desses sistemas levou à classificação das capacidades analíticas em três níveis principais, conforme Sharda, Delen e Turban (2015) [13].

- Análise descritiva: fornece resumos e relatórios baseados em dados passados;
- Análise preditiva: utiliza técnicas como regressão, árvores de decisão e redes neurais para prever tendências ou comportamentos futuros;
- Análise prescritiva: recomenda ações ideais com base em simulações e otimizações matemáticas.

Essas abordagens ampliam significativamente o papel do SAD tradicional, permitindo não apenas apoio à decisão, mas também automação de recomendações em sistemas inteligentes. Tais avanços são particularmente úteis em ambientes digitais [13].

D. O Algoritmo K-Nearest Neighbors (KNN)

O algoritmo KNN é uma técnica simples e eficaz baseada em instâncias. Ele classifica novas amostras com base na proximidade em relação a exemplos conhecidos, considerando uma métrica de distância, geralmente a euclidiana. Sua intuição e aplicabilidade o tornam amplamente utilizado em contextos como diagnóstico médico, reconhecimento de padrões e sistemas de recomendação [7].

Além de sua simplicidade, o KNN apresenta flexibilidade na modelagem de dados não lineares e boa performance em diversos domínios. No entanto, possui limitações como elevado custo computacional na fase de predição em grandes volumes de dados e a sensibilidade à dimensionalidade dos dados [8], [9].

O algoritmo K-Nearest Neighbors é uma técnica de aprendizado supervisionado amplamente utilizada para tarefas de classificação e regressão. Trata-se de um método baseado em instâncias, no qual a predição de uma nova amostra é realizada a partir da proximidade desta em relação às amostras previamente rotuladas, considerando uma métrica de distância usualmente a distância euclidiana entre os pontos no espaço de características [5].

Cálculo da Distância: Uma etapa fundamental no funcionamento do KNN é o cálculo da distância entre a nova amostra (ponto de consulta) e cada uma das amostras presentes no conjunto de dados. A métrica de distância mais utilizada é a *distância euclidiana*, representada pela seguinte equação: [5]

$$d(p, q) = \sqrt{\sum_{i=1}^n (p_i - q_i)^2} \quad (1)$$

Onde:

- $d(p, q)$ é a distância entre os pontos p (ponto de entrada) e q (ponto do conjunto de dados);
- n é o número de atributos (como energia, carboidratos, proteínas, lipídios e fibras);
- p_i e q_i são os valores do i -ésimo atributo dos pontos p e q .

Na figura 1 mostra de forma ilustrativa como funciona graficamente o cálculo do ponto teste X .

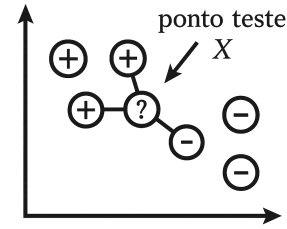


Figura 1. Representação gráfica do ponto de teste X .

Seleção dos k Vizinhos Mais Próximos: Após calcular a distância entre o ponto de entrada x e todos os pontos do conjunto de dados D , o algoritmo seleciona os k vizinhos mais próximos, ou seja, os k pontos com as menores distâncias.

Formalmente, define-se o conjunto dos vizinhos S_x como:

$$S_x \subseteq D \text{ tal que } |S_x| = k \quad (2)$$

$$\forall (x', y') \in D \setminus S_x, \text{dist}(x, x') \geq \max_{(x'', y'') \in S_x} \text{dist}(x, x'') \quad (3)$$

Isso significa que todos os pontos fora de S_x estão a uma distância maior ou igual ao mais distante dos pontos dentro de S_x .

Regra de Decisão: O KNN toma a decisão final com base nos k vizinhos mais próximos:

- Classificação: atribui ao ponto de entrada a classe mais frequente entre os vizinhos. Formalmente:

$$h(x) = \text{mode} \{y'' : (x'', y'') \in S_x\} \quad (4)$$

Onde $h(x)$ é o rótulo predito para o ponto x , e $\text{mode}(\cdot)$ retorna o valor mais comum.

- Regressão: retorna a média dos valores das saídas dos vizinhos.

E. Ferramentas e Tecnologias

O sistema FOI desenvolvido utilizando um conjunto de ferramentas e tecnologias que contribuem para a eficiência, organização e a qualidade do projeto.

1) *Python*: Linguagem de programação de alto nível, conhecida por sua simplicidade e ampla aplicação em sistemas inteligentes e automação [17].

2) *React Native*: Framework para desenvolvimento de interfaces móveis e web com um único código-fonte, garantindo responsividade e agilidade [18].

3) *FastAPI*: Framework web moderno para construção de APIs em Python, focado em alto desempenho e uso de anotações de tipo. A FastAPI mapeia funções Python para rotas HTTP, realiza a validação automática dos dados de entrada por meio de modelos (Pydantic) e converte as respostas em JSON, além de gerar documentação interativa da API (OpenAPI/Swagger). No contexto deste trabalho, a FastAPI é utilizada para expor os endpoints que acessam o banco de dados nutricional e o algoritmo *KNN*, permitindo que o aplicativo cliente consuma esses serviços de forma organizada e padronizada [26].

4) *Expo*: Plataforma e conjunto de ferramentas para desenvolvimento com React Native que simplifica o setup, o ciclo de desenvolvimento e a publicação. Oferece CLI, servidor de desenvolvimento, gerenciamento de ativos, plugins nativos (*config plugins*), builds e distribuição com *EAS* e suporte a atualizações *OTA* para Android, iOS e Web [19].

5) *JavaScript*: Linguagem de programação interpretada, multi-paradigma e orientada a eventos, base do desenvolvimento web moderno. Oferece módulos ES, tipagem dinâmica, suporte a programação assíncrona com *Promises* e *async/await*, e um ecossistema amplo de bibliotecas via npm, sendo executada em navegadores e no ambiente Node.js [25].

6) *HTML e CSS*: *HTML (HyperText Markup Language)* define a estrutura das páginas; *CSS (Cascading Style Sheets)* permite personalização do design [23], [24].

7) *Trello*: Plataforma de gerenciamento de tarefas baseada em quadros e listas, que auxilia no controle e acompanhamento do progresso do projeto [20].

8) *GitHub*: Sistema de controle de versões que registra alterações no código, permitindo rastreamento, colaboração e reversão segura de mudanças [21].

9) *SQLite*: Banco de dados leve e embutido, ideal para armazenar localmente informações de forma rápida e estruturada, sem necessidade de servidor externo [22].

III. TRABALHOS CORRELATOS

Dois estudos recentes servem como base para contextualizar a proposta deste trabalho, especialmente por também abordar sistemas computacionais aplicados à nutrição e à personalização alimentar com o apoio de técnicas de inteligência artificial.

A. Knowledge-based Decision Support System for recommending safe recipes to individuals with dysphagia

O trabalho desenvolvido por Spoladore et al. (2024) propõe um Sistema de Apoio à Decisão (SAD) baseado em estrutura

de conhecimento clínico, voltado à recomendação de receitas seguras para pacientes com disfagia – condição clínica que compromete a deglutição e pode levar à desnutrição. O sistema foi construído em colaboração com especialistas em otorrinolaringologia, neurologia e nutrição, e incorpora conhecimento clínico formalizado por meio de modelo organizado de informações clínicas, utilizando as escalas DOSS (*Dysphagia Outcome and Severity Scale*) que avalia funcionalmente o grau de disfagia e a segurança da via oral e PAS (*Penetration-Aspiration Scale*) que mede o risco de penetração ou aspiração de alimentos nas vias aéreas para classificar a severidade da disfagia e adequar as consistências alimentares às limitações dos pacientes. Para classificar a severidade da disfagia e adequar as consistências alimentares às limitações dos pacientes [3].

Este trabalho contribuiu para a pesquisa ao demonstrar a viabilidade de integrar conhecimento clínico com tecnologias digitais para gerar recomendações alimentares seguras. Serviu de base conceitual para reforçar a proposta de personalização nutricional, validando a importância de adaptar sugestões alimentares de cada paciente com abordagem baseada em aprendizado de máquina.

B. Aplicação do Classificador K-Nearest Neighbors (KNN) na Área da Saúde: Relação Cintura-Quadril e Pressão Arterial

O trabalho de Passos et al. (2021) teve como objetivo aplicar o algoritmo *K-Nearest Neighbors KNN* para classificar indivíduos com base em dois indicadores importantes de saúde: a relação cintura-quadril e os níveis de pressão arterial. A proposta visou verificar a eficácia do *KNN* como classificador em um cenário real de aplicação na área médica, onde essas variáveis são comumente associadas ao risco de doenças cardiovasculares [16].

Este trabalho contribuiu de forma prática para o desenvolvimento do sistema apresentado, ao demonstrar como o algoritmo *KNN* pode ser aplicado em contextos relacionados à saúde. A forma como o estudo foi estruturado e os dados foram utilizados ajudaram a refletir sobre a possibilidade de empregar esse tipo de abordagem em sistemas nutricionais.

IV. METODOLOGIA

A construção do sistema seguiu os princípios da engenharia de software, adotando uma abordagem ágil, mais especificamente o framework *Scrum* [15]. A escolha por essa metodologia se deu devido à sua capacidade de promover entregas incrementais, curtas e contínuas, aliada à flexibilidade para lidar com mudanças nos requisitos ao longo do tempo.

Este projeto utiliza uma adaptação do *Scrum*. A principal mudança são reuniões diárias e dá lugar a uma reunião semanal com o orientador, usada como ponto de verificação para revisar o avanço, reordenar prioridades do *backlog* e tratar impedimentos. Os ciclos têm duração flexível de 1 a 2 semanas, definida conforme a complexidade das entregas e o calendário acadêmico. As demais práticas e artefatos são mantidos, com plano de ciclo, análise e retrospectiva, ocorrendo no início e no fim de cada ciclo [15].



Figura 2. Adaptação do framework Scrum para o desenvolvimento do projeto.

A Figura 2, representa visualmente a adaptação do Scrum ao contexto de desenvolvimento desse trabalho. O ciclo inicia com a definição do *backlog do produto*, que reúne os requisitos do sistema, revisão e desenvolvimento. A cada ciclo semanal, Uma nova funcionalidade é desenvolvida e testada, Que ao final de cada *sprint* são realizadas revisões do ciclo e ajustes no *backlog*. Esse processo se repete de forma interativa até a conclusão do trabalho.

Durante o processo, foi utilizada a ferramenta *Trello* para gerenciamento visual das tarefas e controle do progresso das ciclos. E o versionamento do código foi mantido com o *Git*, assegurando rastreabilidade e facilitando a colaboração e controle das versões.

A. Descrição do Sistema

O presente trabalho desenvolveu um Sistema de Apoio à Decisão para auxiliar nutricionistas na substituição e recomendação de alimentos com base nos valores de nutrientes semelhantes. A Figura 3 apresenta uma visão geral do funcionamento do sistema, destacando a interação entre o nutricionista, a interface web, o servidor de aplicação e a base de dados de composição nutricional.

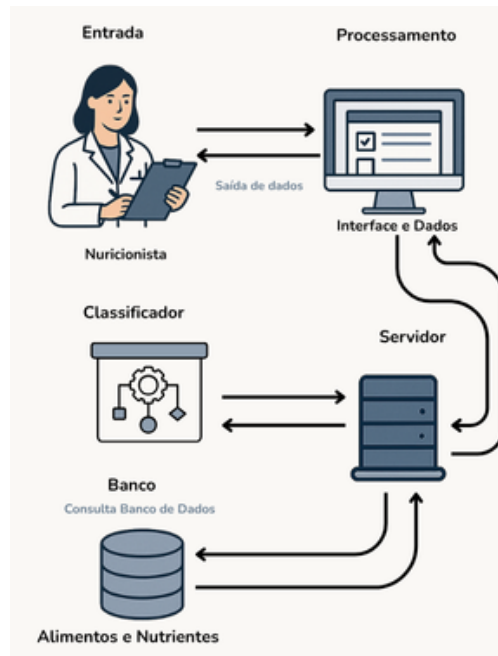


Figura 3. Visão do funcionamento do sistema proposto

O ponto de partida é a entrada de dados pelo profissional de saúde. Na interface web, o nutricionista pode localizar o alimento de referência de duas formas: digitando diretamente seu nome na barra de pesquisa (por exemplo, “carne bovina, coxão mole sem gordura”) ou navegando pelas categorias do sistema, como “Carnes e derivados”, “Cereais e derivados”, entre outras. Uma vez encontrado o alimento desejado, o usuário o seleciona e, em seguida, escolhe as características nutricionais que deseja preservar na substituição, como energia (kcal), proteínas, carboidratos, lipídios, fibras e demais micronutrientes relevantes ao planejamento da dieta.

Após a seleção do alimento base e dos nutrientes de interesse, o nutricionista, por meio da interface, solicita o cálculo de similaridade, enviando esses dados ao servidor, que aciona o módulo classificador. Esse módulo implementa o algoritmo *K-Nearest Neighbors*, responsável por calcular a similaridade nutricional entre o alimento informado e os demais alimentos presentes na base de dados, previamente alimentada com informações da Tabela Brasileira de Composição de Alimentos. O algoritmo calcula as distâncias no espaço dos nutrientes escolhidos e seleciona os *k* alimentos mais próximos, ordenando-os de forma crescente pela distância euclidiana, isto é, dos mais semelhantes para os menos semelhantes.

Em seguida, o servidor retorna à interface uma lista ordenada de alimentos considerados nutricionalmente similares ao alimento de referência. Essa lista é exibida ao nutricionista em uma página web, contendo os valores nutricionais detalhados e uma indicação de similaridade, permitindo comparar rapidamente as opções sugeridas. As recomendações são apresentadas como sugestões de substituição, cabendo sempre ao profissional avaliar, à luz do contexto clínico e das preferências do paciente, quais alternativas serão efetivamente

Tabela II
HISTÓRIAS DE USUÁRIO ORGANIZADAS POR ÉPICO

Épico	ID	Descrição da História de Usuário
Entrada de Dados	HU1.1	Como nutricionista, quero informar um alimento de referência, para que o sistema possa iniciar a recomendação.
	HU1.2	Como nutricionista, quero indicar prioridades nutricionais (ex: proteínas ou fibras), para personalizar as recomendações.
	HU1.3	Como nutricionista, quero acessar o banco de dados manualmente.
Processamento com KNN	HU2.1	Como nutricionista, quero que o sistema utilize um algoritmo de similaridade nutricional para sugerir alimentos com perfil semelhante ao alimento de entrada.
	HU2.2	Como sistema, quero acessar o banco de dados para calcular a similaridade nutricional entre alimentos.
	HU2.3	Como sistema, devo ordenar as sugestões do alimento mais semelhantes por pontuação.
Exibição de Resultados	HU3.1	Como nutricionista, quero visualizar a lista de alimentos recomendados com seus dados nutricionais detalhados.
	HU3.2	Como nutricionista, quero acessar informações detalhadas de cada alimento para compará-los.
Interface e Usabilidade	HU4.1	Como nutricionista, quero uma interface responsiva para computador.
	HU4.2	Como nutricionista, quero que os campos do formulário sejam claros e exemplificados, para evitar erros na entrada de dados.
	HU5.3	Como nutricionista, quero que os dados nutricionais sejam apresentados de forma organizada em tabela, para facilitar comparações.

adotadas. Caso deseje, o nutricionista ainda pode recorrer à busca manual, acessando diretamente o banco de dados para explorar outras possibilidades de alimentos.

B. Requisitos do Sistema

Os requisitos do sistema foram estruturados no formato de histórias de usuário, elaboradas considerando a experiência prática do nutricionista no processo de prescrição alimentar, analisando os dados nutricionais da Tabela TACO, os fluxos de interação do sistema e as funcionalidades essenciais ao contexto.

Essas histórias foram agrupadas em épicos, de forma a reunir funcionalidades relacionadas em torno de um mesmo objetivo dentro do fluxo de uso da aplicação. Assim, foram definidos quatro épicos principais: *Entrada de Dados*, *Processamento com KNN*, *Exibição de Resultados* e *Interface e Usabilidade*, conforme apresentado na Tabela II.

C. Projeto

Nesta seção é apresentado o diagrama de domínio, referenciado pela Figura 4. As classes e seus vínculos refletem a estrutura dos dados utilizada pelo *backend* para executar o *KNN* e montar a resposta enviada à interface.

O modelo explicita o relacionamentos e os atributos relevantes para a aplicação: uma *Recomendação* está sempre associada a um único *Alimento* base e agrega um conjunto de *Alimentos Recomendados*. Cada item recomendado, por sua vez, está ligado a exatamente uma *Recomendação* e a um único *Alimento* da base de dados, que fornece a descrição textual, a categoria e os nutrientes usados no cálculo de similaridade. Essa visão de alto nível serve como mapa para a leitura

das subseções a seguir, onde cada classe é descrita de forma objetiva.

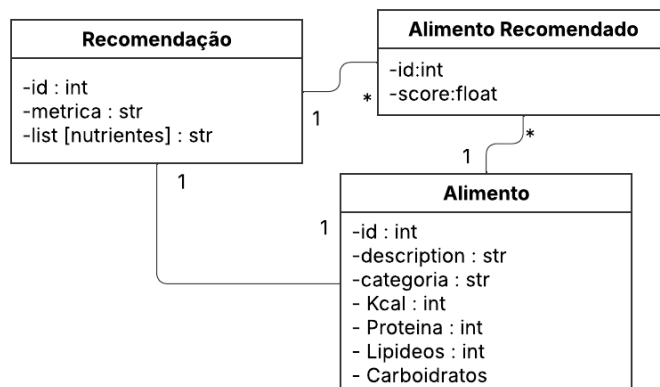


Figura 4. Diagrama de classes do domínio do sistema proposto

Recomendação: Conjunto de sugestões geradas a partir de um alimento base. A entidade armazena um identificador da recomendação, a *métrica* de similaridade selecionada e a lista de *nutrientes* considerados no cálculo. A partir desses parâmetros, a recomendação agrupa vários *Alimentos Recomendados* associados ao alimento de referência.

Alimento Recomendado: Item sugerido pelo sistema como alternativa ao alimento base. Cada registro guarda o *id* do alimento da base e o seu *score* de similaridade, em escala percentual, calculado a partir dos nutrientes informados na recomendação. Os demais dados do alimento (descrição, categoria e nutrientes) são obtidos via relacionamento com a entidade *Alimento*.

Alimento: Define a estrutura de um alimento com base nos dados da Tabela Brasileira de Composição de Alimentos. Contém atributos como *id*, *description*, *categoria* e nutrientes numéricos, por exemplo *Kcal*, proteína, lipídeos e carboidratos, entre outros presentes na tabela original. Esses valores servem de entrada para a comparação nutricional realizada pelo algoritmo de similaridade.

D. Implementação

A implementação seguiu o pipeline da construção da base de dados, implementação do algoritmo *KNN*, desenvolvimento da interface web. A seguir, cada componente é descrito com os trechos essenciais em pseudocódigo.

1) *Construção da base de dados*: A Tabela Brasileira de Composição de Alimentos foi integrada ao sistema a partir de um arquivo em formato *JSON - JavaScript Object Notation*, obtido do repositório GitHub de Marcelo Santo [14]. Após validar que os dados correspondiam à tabela original, o arquivo foi convertido para um banco de dados *SQLite* por meio de um conversor implementado em Python, responsável por transformar cada objeto do arquivo *JSON* em uma linha de tabela relacional. Cada um destes objetos do arquivo contém os atributos do alimento no formato chave-valor. Valores como “NA”, “Tr” e campos vazios demonstrado no Listing 1, foram convertidos em *NULL* e armazenados em colunas numéricas do *SQLite*. O banco é composto por uma única tabela, com as colunas *Id*, *description*, *category*, *energy-kcal*, etc, que foram traduzidos e convertidos para os valores decimais conforme a tabela original.

```
1 {
2   "id": 351,
3   "description": "Carne, bovina, cox o
4   mole, sem gordura, cozido",
5   "category": "Carnes e derivados",
6   "humidity_percents": 58.04333333333333,
7   "energy_kcal": 218.6751,
8   "energy_kj": 914.9366184,
9   "protein_g": 32.38333333333333,
10  "lipid_g": 8.913333333333333,
11  "cholesterol_mg": 84.15,
12  "carbohydrate_g": 0,
13  "fiber_g": "NA",
14  "manganese_mg": "Tr",
15  ...

```

Listing 1. Exemplo de trecho do arquivo TACO.json

A normalização dos valores numéricos é realizada no conversor em Python, que converte diferentes formatos de entrada (como strings com vírgula decimal) para objetos *Decimal* e aplica o arredondamento adequado por nutriente, conforme mostrado no Listing 2.

```
1 def parse_decimal_loose(x: Any) -> Decimal
2   | None:
3   if isinstance(x, (int, float, Decimal)):
4   :
5   s = str(x)
6   elif isinstance(x, str):
7   s = x.strip()
8   if not s or s in {"NA", "Tr"}:

```

```
7         return None
8         if "," in s and "." not in s:
9             s = s.replace(".", "").replace(
10                ",", ".")
11         else:
12             return None
13         try:
14             return Decimal(s)
15         except InvalidOperation:
16             return None
17 def round_half_up(d: Decimal, places: int)
18   -> float:
19     q = Decimal("1") if places == 0 else
20     Decimal("0." + "0"*(places-1) + "1")
21     return float(d.quantize(q, rounding=
22        ROUND_HALF_UP))

```

Listing 2. Normalização de valores numéricos para decimais no conversor

2) *Implementação do algoritmo KNN*: O *backend* é implementado em *Python* e exposto como serviço web com *FastAPI*, que organiza os endpoints HTTP, valida os parâmetros de entrada e integra com o banco *SQLite*. A similaridade nutricional é calculada por um módulo próprio de *KNN*, que trabalha com o banco de dados para a informações de cada alimento. O fluxo consiste em: selecionar as colunas nutricionais, carregar o vetor do alimento base, montar a matriz de candidatos e chamar a função de *KNN* que devolve a lista pontuada, conforme ilustrado na Listing 3.

```
1 feat_cols = auto ? repo.numeric_cols() :
2   parse(cols)
3 base_vec = repo.vector(food_id, feat_cols)
4 cands = repo.matrix(feat_cols, same_cat,
5   food_id)
6 scored = knn_compute(base_vec, cands, k)
7 return { base_id: food_id, items: scored }

```

Listing 3. Orquestração do endpoint KNN

a) *Pré-processamento*: Antes de aplicar o *KNN*, os dados passam por um pré-processamento em *Python*, que combina imputação por média e normalização *z-score*. Esse módulo calcula estatísticas por coluna (média e desvio padrão), substitui valores ausentes ou inválidos (como *NULL*) pela média da coluna e, em seguida, transforma cada atributo para uma escala normalizada com média zero e desvio padrão igual a um. Esse procedimento garante que nutrientes medidos em escalas distintas (por exemplo, miligramas de sódio e quilocalorias) contribuam de forma equilibrada na distância final. Conforme demonstra o Listing 4.

```
1 stats = compute_stats([base] + cands)
2 base = zscore(impute(base, stats))
3 cands = map(cands, v => zscore(impute(v,
4   stats)))
5 w = ones(len(base))

```

Listing 4. Imputação e normalização

b) *Cálculo da distância*: O cálculo de similaridade é feito via distância euclidiana ponderada, implementada no *backend* em *Python* sem dependência de ferramentas externas de aprendizado de máquina. Para cada alimento candidato, o

código percorre o vetor de atributos já normalizados, aplica os pesos (armazenados em w) e acumula a soma dos quadrados das diferenças entre o alimento base e o candidato. Ao final, a raiz quadrada dessa soma produz a distância euclidiana, que é usada para ordenar os vizinhos mais próximos. A Listing 5 ilustra o cálculo de distância e prepara a lista de resultados.

```
1 raw = []
2 for v in cand:
3     dist = sqrt(sum(w[i] * (base[i]-v[i])^2))
4     raw.append((v.id, v.desc, v.cat, dist))
5
6 scored = pct_from_dist(raw)
7 return top_k(scored, k)
```

Listing 5. Distâncias e seleção dos top- k vizinhos

c) *Conversão para pontuação (0–100)*: Para tornar a saída mais esclarecida pelo usuário final, as distâncias são convertidas em uma pontuação de similaridade na escala de 0 a 100. Esse módulo, também implementado em *Python*, identifica a menor e a maior distância observadas e aplica uma normalização linear: a menor distância recebe pontuação mais alta, e a maior distância recebe pontuação próxima de zero. Em seguida, o valor é limitado ao intervalo $[0, 100]$ e arredondado para uma casa decimal, produzindo a pontuação exibida na interface. A Listing 6 apresenta essa normalização.

```
1 dmin, dmax = minmax(dists)
2 pct = (dmax - d) / (dmax - dmin) * 100
3 pct = round(clamp(pct, 0, 100), 1)
```

Listing 6. Normalização das distâncias em pontuação de 0 a 100

3) *Desenvolvimento da interface web*. : A interface do sistema foi desenvolvida em *React*, executada em ambiente *Expo* e estruturada com componentes do *React Native* renderizados no navegador por meio do *React Native Web*. As telas são organizadas em uma única página, controlada por estados internos, e incluem: página inicial, seleção por categorias, detalhe do alimento de referência, página de resultados com recomendações e uma página “Sobre”. A comunicação com o *backend* em *FastAPI* é feita por meio de requisições *HTTP* que retornam dados em formato *JSON*, posteriormente exibidos em listas, cartões e tabelas com os alimentos similares e suas pontuações.

Para a camada visual, são utilizados elementos *HTML* (campo de busca, botões, contêineres e tabelas) combinados com estilos e variáveis *CSS*, garantindo um layout limpo e consistente. O sistema suporta tema claro e escuro por meio de um seletor na barra de navegação: ao alternar o tema, são atualizadas variáveis globais de cor e a preferência do usuário é persistida, mantendo a aparência entre sessões. A responsividade é obtida com *flexbox* e contêineres de largura fluida, permitindo o uso confortável tanto em monitores quanto em dispositivos móveis.

Do ponto de vista de usabilidade, a interface implementa um campo de busca com *debounce* para evitar requisições excessivas, o sistema valida se há pelo menos um nutriente selecionado; caso contrário, é exibido um alerta no meio da tela para orientar o usuário a escolher as colunas desejadas.

Esses recursos, aliados à navegação simples entre “Início” e “Sobre”, contribuem para uma interação mais clara e agradável para o nutricionista.

V. RESULTADOS E DISCUSSÃO

A. Tela Inicial

Na parte superior, há uma barra fixa com o nome fictício da aplicação (*NutriFinder*) e três opções: *Início*, que permite retornar à tela principal; *Sobre*, que exibe informações adicionais sobre a ferramenta; e o seletor de tema *Escuro*, que alterna entre os modos claro e escuro. Logo abaixo da barra, são exibidos o título “Tabela de Busca de Similaridade Nutricional” e um texto descritivo indicando que a Tabela TACO é utilizada como referência.

Em seguida, há um campo de pesquisa em que o usuário pode digitar diretamente o alimento desejado. Alternativamente, ele pode utilizar os cards de categoria mostrados abaixo, como “Todos”, “Carnes e derivados”, “Frutas e derivados”, entre outros, para selecionar um grupo específico de alimentos. Essa interface é ilustrada na Figura 5.

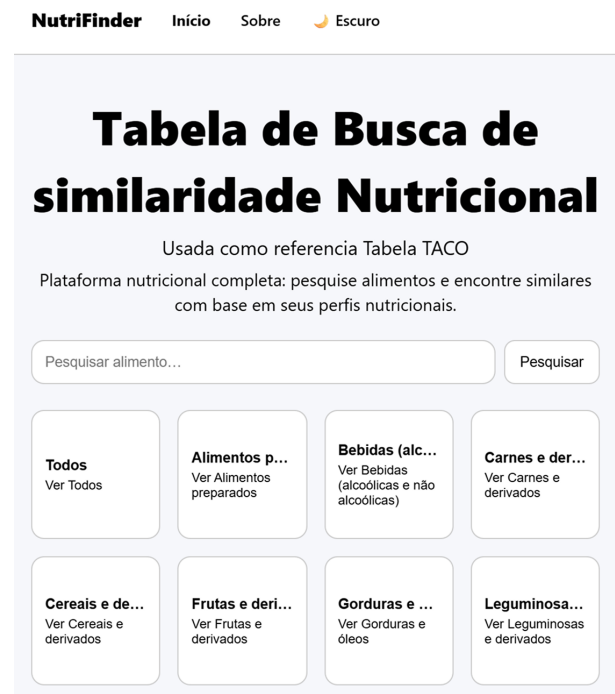


Figura 5. Tela inicial com campo de busca e seleção de alimentos baseado em categorias nutricionais.

B. Seleção de alimento

Quando o usuário escolhe uma categoria, como *Carnes e derivados*, o usuário é direcionado para uma tela que lista todos os alimentos dessa categoria. Cada alimento é apresentado com seu nome, energia (kcal), carboidratos e proteínas, além da opção “Saiba mais”, que permite acessar informações adicionais e prosseguir com o cálculo. Essa etapa é ilustrada na Figura 6.

NutriFinder Início Sobre Escuro

← Voltar

Carnes e derivados

coxão Pesquisar

Valores nutricionais com base em 100g do alimento.

Carne, bovina, coxão duro, sem gordura, cozido

Energia (kcal)	Carboidratos	Proteínas
217.0	0.00	31.90

Saiba mais →

Carne, bovina, coxão duro, sem gordura, cru

Energia (kcal)	Carboidratos	Proteínas
148.0	0.00	21.50

Saiba mais →

Carne, bovina, coxão mole, sem gordura, cozido

Energia (kcal)	Carboidratos	Proteínas
219.0	0.00	32.40

Saiba mais →

Figura 6. Seleção de alimento.

C. Função selecionar nutrientes

NutriFinder Início Sobre Escuro

← Voltar

Carne, bovina, coxão mole, sem gordura, cru — Carnes e derivados

Tabela nutricional

Umidade (%)	Energia (kcal)
68.60	169.0
Energia (kJ)	Proteína (g)
707.0	21.20
Lipídeos (g)	Colesterol (mg)
8.70	84.00

Escolha quantos alimentos similares deseja ver : 10

Mesma categoria: não Nutrientes selecionados: 3 **Calcular similaridade**

Figura 7. Seleção de nutrientes.

Na Figura 7 é selecionada o exemplo de coxão mole, sem gordura, cru de carnes e derivados, com a seleção de Kcal, Proteína e Lipídeos. É exibida uma tabela com todos os valores nutricionais, permitindo que o usuário selecione aqueles que deseja utilizar no cálculo. Abaixo, há uma barra fixa em que é possível definir quantos alimentos similares serão exibidos, se o cálculo será restrito à mesma categoria do alimento base e acionar o botão de calcular similaridade, conforme ilustrado.

D. Resultado da pesquisa

Ao acionar o botão *Calcular similaridade*, o alimento de referência é exibido na parte superior da tela, juntamente

com os nutrientes selecionados (por exemplo, energia, proteína e lipídeos). No canto superior direito desse card, a opção *Exibir tabela completa* (permite visualizar todos os nutrientes disponíveis). Abaixo, é apresentada uma tabela com os alimentos mais similares, indicando para cada um a descrição, a categoria, os valores dos nutrientes escolhidos e a pontuação de similaridade, o que auxilia o usuário a identificar quais opções estão mais próximas do alimento base.

No exemplo apresentado na Figura 8, a substituição de *carne, bovina, coxão mole, sem gordura, cru* com os nutrientes selecionados: 169 kcal, 21,20 g de proteína e 8,70 g de lipídeos, resulta na recomendação do alimento *Porco, bisteca, crua*, da categoria *Carnes e derivados*, que atinge 100 % de pontuação ilustrada como a mais próxima e apresenta, para os nutrientes selecionados, 164 kcal, 21,50 g de proteína e 8,00 g de lipídeos, com 98,10% de pontuação ilustrada como a mais distante e apresenta *Salmão, sem pele, fresco, cru* com os nutrientes selecionados: 170 kcal, 19,30 g de proteína e 9,70 g de lipídeos.

NutriFinder Início Sobre Escuro

← Voltar

Carne, bovina, coxão mole, sem gordura, cru — Carnes e derivados Exibir tabela completa

Energia (kcal)	Proteína (g)	Lipídeos (g)
169.0	21.20	8.70

Valores nutricionais com base em 100g do alimento dos similares.

Descrição	Categoria	Energia (kcal)	Proteína (g)	Lipídeos (g)	Pontuação
Porco, bisteca, crua	Carnes e derivados	164.0	21.50	8.00	100.00
Carne, bovina, miolo de alcatra, sem gordura, cru	Carnes e derivados	163.0	21.60	7.80	99.80
Carne, bovina, paleta, com gordura, crua	Carnes e derivados	159.0	21.40	7.50	99.50
Porco, lombo, cru	Carnes e derivados	176.0	22.60	8.80	98.90
Carne, bovina, maminha, crua	Carnes e derivados	153.0	20.90	7.00	98.80
Frango, peito, com pele, cru	Carnes e derivados	149.0	20.80	6.70	98.40
Salmão, sem pele, fresco, cru	Pescados e frutos do mar	170.0	19.30	9.70	98.10

Figura 8. Resultados.

VI. CONCLUSÃO

Este trabalho partiu do desafio de apoiar nutricionistas na substituição de alimentos de forma rápida e fundamentada em dados. O objetivo foi construir um sistema que, a partir da tabela TACO, que identifique alimentos com perfis nutricionais

semelhantes e apresente opções de substituição. O desenvolvimento seguiu Scrum adaptado, com requisitos estruturados em histórias de usuário.

Foi implementado um sistema funcional com *backend* em Python e dados retirados da tabela brasileira de composição de alimentos importada para SQLite, o algoritmo *KNN* utiliza a distância euclidiana para fazer suas análises, considerando apenas as colunas escolhidas e que foram normalizadas usando o método z-score. As similaridades são apresentadas como *score* de 0 a 100, o que facilita a leitura. A interface permite busca por texto e por categorias, seleção de nutrientes prioritários e visualização das alternativas sugeridas.

O sistema permite o objetivo de receber um alimento de referência, processa as variáveis nutricionais, calcula a proximidade euclidiana e retorna um ranking de substituições com justificativa transparente (colunas usadas, métrica e pontuação). O profissional consegue, em poucos passos, comparar opções e decidir. A solução reduz tempo de análise, padroniza critérios e dá suporte a decisões. A apresentação em *score* torna a recomendação compreensível para o usuário demonstrando, contribuindo para atendimentos mais rápidos.

A versão atual ainda utiliza apenas a métrica euclidiana e depende da integridade e qualidade dos dados da TACO. Não há personalização por paciente, não é possível selecionar apenas uma tabela da escolha do nutricionista para visualizar substituição. Exemplo: selecionar "Coxão mole" e optar por apenas certas categorias sejam selecionadas como: "Leguminosas e derivados" para pacientes que não ingerem carne.

Como próximos passos, propõe-se concentrar esforços em três frentes principais: realizar testes com nutricionistas e estudos de validação clínica das substituições sugeridas; incluir o cálculo por porção em gramas, permitindo estimar automaticamente quantidades equivalentes entre alimentos (por exemplo, a quantidade de batata necessária para substituir 200 g de arroz com perfil nutricional semelhante); e investigar novas métricas de similaridade; pesquisar outras tabelas com informações nutricionais para melhor pesquisa de nutrientes.

REFERÊNCIAS

- [1] C. Brito, "Dietas restritivas podem causar problemas de saúde e ganho de peso," *Jornal da USP*, 05-Jan-2017. [Online]. Available: <https://jornal.usp.br/atualidades/dietas-restritivas-podem-causar-problemas-de-saude-e-ganho-de-peso/>. Accessed: Mar. 7, 2025.
- [2] T. Shimizu, *Decisão nas Organizações*, 3ª ed., São Paulo: Atlas, 2010.
- [3] D. Spoladore et al., "A Knowledge-Based Decision Support System for Recommending Safe Recipes to Individuals with Dysphagia," *Computers in Biology and Medicine*, 2024.
- [4] R. Ali et al., "Intelligent Decision Support Systems—An Analysis of Machine Learning and Multicriteria Decision-Making Methods," *Applied Sciences*, vol. 13, no. 22, p. 12426, 2023.
- [5] IBM, "K-Nearest Neighbors (KNN)," IBM Think. [Online]. Available: <https://www.ibm.com/br-pt/think/topics/knn>. Accessed: Mar. 7, 2025.
- [6] NEPA/UNICAMP, *Tabela Brasileira de Composição de Alimentos – TACO*, 4ª ed. revisada e ampliada, Campinas: UNICAMP, 2011. [Online]. Available: <https://www.unicamp.br/nepa/taco>. Accessed: Apr. 17, 2025.
- [7] B. Venkateswarlu and R. Gangula, "Exploring the Power and Practical Applications of K-Nearest Neighbours (KNN) in Machine Learning," *Journal of Computer Allied Intelligence*, vol. 2, no. 1, pp. 8–15, 2024.
- [8] A. Acito, "Introdução ao KNN e seus desafios em ambientes de alta dimensionalidade," *Revista Científica de Aprendizado de Máquina*, 2023.
- [9] A. Biswas et al., "Comparative analysis of KNN performance under curse of dimensionality," *IEEE Transactions on Pattern Analysis*, 2023.
- [10] L. K. Mahan and J. L. Raymond, *Krause: Alimentos, Nutrição e Dietoterapia*, 14ª ed., São Paulo: Elsevier, 2018.
- [11] J. A. Bittencourt, *Nutrição e Saúde: Como Fazer Escolhas Sensatas em Dieta e Nutrição*, 6. ed., São José dos Campos: J. A. Bittencourt, 2018.
- [12] A. Philippi, *Técnica Dietética: Seleção e Preparação dos Alimentos*, 2. ed., Barueri: Manole, 2008.
- [13] R. Sharda, D. Delen and E. Turban, *Business Intelligence and Analytics: Systems for Decision Support*, 10th ed., Pearson Education, 2015.
- [14] M. Santos, *Tabela Brasileira de Composição de Alimentos em formato JSON (TACO.json)*, GitHub, 2025. [Online]. Available: https://github.com/marcelosanto/tabela_taco/blob/main/TACO.json. Accessed: 17 nov, 2025.
- [15] I. Sommerville, *Engenharia de Software*, 9ª ed., São Paulo: Pearson Prentice Hall, 2011.
- [16] R. P. Passos, L. F. Sílio, B. N. Lima et al., "Aplicação do Classificador K-Nearest Neighbors (KNN) na Área da Saúde: Relação Cintura-Quadril e Pressão Arterial," in *Congresso de Ciências do Desporto e da Saúde do CPAQV*, Maringá: CPAQV, 2021.
- [17] Python Software Foundation, "Python Programming Language." [Online]. Available: <https://www.python.org>.
- [18] Meta Platforms, "React Native – Build Mobile Apps with React." [Online]. Available: <https://reactnative.dev>.
- [19] Expo, "Guides: Overview — Expo Documentation." [Online]. Available: <https://docs.expo.dev/guides/overview/>. Accessed: Nov. 7, 2025.
- [20] Atlassian, "Trello – Project Management Tool." [Online]. Available: <https://trello.com>.
- [21] GitHub, "Project Management – GitHub Projects." [Online]. Available: <https://github.com/features/project-management>.
- [22] SQLite Consortium, "SQLite: Embedded Database Engine." [Online]. Available: <https://www.sqlite.org>.
- [23] WHATWG, "HTML Living Standard." [Online]. Available: <https://html.spec.whatwg.org>.
- [24] W3C, "Cascading Style Sheets (CSS)." [Online]. Available: <https://www.w3.org/Style/CSS/>.
- [25] MDN Web Docs, "JavaScript." [Online]. Available: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>.
- [26] RAMÍREZ, Sebastián. *FastAPI: modern, fast (high-performance), web framework for building APIs with Python*. 2018. Disponível em: <https://fastapi.tiangolo.com/>. Acesso em: 17 nov. 2025.