A B A K Ó S
PUC Minas Instituto de Ciências Exatas e Informática

Desenvolvimento de um sistema para criação de planos alimentares baseado em orçamento e preferência de refeições*

João Pedro Teixeira Dias[1]
Magali Rezende Gouvêa Meireles[2]
Octavio Augusto Pereira Martins[3]

Resumo

O presente trabalho descreve o desenvolvimento de um aplicativo móvel dedicado ao âmbito nutricional, oferecendo a capacidade de personalizar um plano de alimentação para qualquer pessoa. O objetivo é simplificar o processo de planejamento, economia e aderência a uma dieta. Com base nas informações fornecidas pelo usuário, o aplicativo permite a substituição de refeições e alimentos, respeitando as diretrizes da dieta. Além disso, ele oferece a flexibilidade de ajustar o plano alimentar de acordo com preferências e restrições individuais. O aplicativo disponibiliza uma lista abrangente de alimentos e refeições, permitindo ao usuário realizar substituições. Quanto aos custos, cada item na lista é acompanhado pelo seu preço, conforme fornecido pelas lojas ou supermercados, ajudando o usuário a tomar decisões financeiramente conscientes.

Palavras-chave: Nutricional, Planejamento, Praticidade, Plano Alimentar, Gastos, Orçamento, Preferência.

*Artigo apresentado à Revista Abakos

[1]Bacharelando em Sistemas de Informação pela PUC Minas, Brasil, 1272244@sga.pucminas.br
[2]Doutora em Ciência da Informação pela UFMG, Brasil, magali@pucminas.br
[3]Bacharelando em Sistemas de Informação pela PUC Minas, Brasil, oapmartins@sga.pucminas.br

Abstract

The present work describes the development of a mobile application dedicated to the nutritional field, offering the ability to customize a meal plan for anyone. The goal is to simplify the process of planning, saving, and adhering to a diet. Based on the information provided by the user, the application allows for the substitution of meals and foods while adhering to the dietary guidelines. Furthermore, it provides the flexibility to adjust the meal plan according to individual preferences and restrictions. The application offers a comprehensive list of foods and meals, enabling users to make substitutions. In terms of costs, each item on the list is accompanied by its price as provided by stores or supermarkets, assisting users in making financially conscious decisions.

Keywords: Nutritional, Planning, Practicality, Diet Plan, Expenses, Budget, Preference.

1 INTRODUÇÃO

Aplicações que buscam auxiliar o usuário em seus objetivos e necessidades de saúde, estão sendo cada vez mais populares e reconhecidas pelos usuários e também pelos profissionais da saúde. Dentre esses aplicativos, os voltados à nutrição são reconhecidos por auxiliar usuários a manter um estilo de vida saudável (WANGLER; JANSKY, 2021). Este trabalho visa unir os conhecimentos de desenvolvimento de aplicações, e os métodos nutricionais de construção de plano alimentar, visando auxiliar o usuário final a encaixar um plano alimentar ao orçamento disponível para o mesmo.

Diante disso, abre-se uma oportunidade de inovar e trazer ao mercado, uma aplicação que fornece um plano alimentar, onde o usuário pode substituir as próprias refeições ou alimentos, por outras de seu próprio interesse. Caso um item do plano alimentar seja selecionado, é exibida uma lista com possíveis alimentos disponíveis para troca. A troca não é aleatória por qualquer refeição, é realizada com base no horário, alimentos já inclusos na dieta e preço dos alimentos. O valor dos alimentos, e do plano alimentar completo, são exibidos a todo momento no aplicativo, dando ao usuário a possibilidade de ter maior controle em relação aos gastos em sua dieta.

O objetivo geral deste trabalho é o desenvolvimento de um aplicativo que busca facilitar a vida de pessoas que pretendem fazer dieta, dando maior controle sobre os gastos com cada alimento e refeição, oferecendo planos alimentares que possam ser modificados com base nas preferências e condição financeira. Para atingir esse objetivo, estabelecemos os seguintes objetivos específicos:

1. Criação de um sistema onde o usuário consegue informar uma dieta e o horário em que cada refeição deverá ser realizada.

2. Criação de um sistema de trocas de alimentos respeitando as características e macronutrientes da dieta.

3. Criação de um sistema de cálculo de preço, onde o usuário poderá ver o valor da sua dieta.

4. Integração de algoritmo de correspondência de gostos alimentares para sugerir opções de refeições personalizadas.

5. Projeto de interface de usuário intuitiva e amigável que é de fácil navegação e registro de informações.

Estes objetivos específicos foram definidos para garantir que o aplicativo atenda às necessidades dos usuários de maneira eficaz e eficiente, fornecendo controle financeiro e planos alimentares personalizados para facilitar a jornada de dieta de cada indivíduo.

As próximas seções estão organizadas como descrito a seguir. A Seção 2 apresenta o referencial teórico, contendo informações sobre dieta e seus componentes, arquitetura, tecnologias adotadas no sistema, e artigos que foram relacionados com o presente trabalho. A Seção 3

apresenta a metodologia aplicada, o planejamento, a definição das etapas a forma de validação. A seção 4 apresenta o tópico de desenvolvimento da aplicação, onde é informado tudo que foi aplicado para contrução da aplicação. Por fim, a seção 5, apresenta o tópico das considerações finais, onde é feito um resumo geral do trabalho e a conclusão.

2 REFERENCIAL TEÓRICO

As próximas subseções apresentam conceitos relacionados a dieta e seus componentes, arquitetura de sistemas, e tecnologias e ferramentas para desenvolvimento do aplicativo.

2.1 Dieta e seus componentes

Uma dieta é definida por tudo aquilo que um indivíduo se alimenta. É feita uma distribuição de macro e micronutrientes, de modo que, idealmente, cada pessoa terá uma distribuição específica visando atingir um objetivo.

Os carboidratos, proteínas e lipídios, por serem os nutrientes de maior contribuição energética para uma dieta, se enquadram na classe dos macronutrientes. Uma distribuição adequada desses nutrientes é de extrema importância para a nutrição de um paciente. Essa distribuição é feita em relação ao percentual de calorias que o indivíduo necessita. Portanto, indivíduos diferentes, com necessidades e objetivos diferentes, terão diferentes distribuições de macronutrientes em seu plano alimentar.

As vitaminas e minerais são considerados os micronutrientes. Isso acontece pois esses compostos não apresentam relevante potencial calórico e energético para uma dieta. Além disso são necessários para as reações químicas que ocorrem no corpo. Os micronutrientes podem ser encontrados em diversos alimentos, sendo mais abundantes em frutas legumes e verduras (MAGNO et al., 2018).

Uma dieta pode ser definida por tudo que é ingerido por uma pessoa, independente dos objetivos: ganhar massa, perder peso ou realizar a manutenção do peso. Existem diversos tipos de dietas com propósitos diferentes, mas para o presente trabalho, foram abordados os três tipos.

2.2 Arquitetura de Sistemas

A arquitetura adotada durante o desenvolvimento foi o MVC (*Model-View-Controller*). Este padrão de arquitetura foi criado por Trygve Reenskaug, em 1979. Com ele, é possível realizar a divisão do projeto em camadas, viabilizando a independência e reutilização de componentes, e o desenvolvimento de um código organizado e enxuto (SILVA, 2012).

A composição do MVC é feita pelas camadas de Modelo, Visão e Controle. O Modelo é

responsável pelo acesso e manipulação dos dados dentro da aplicação. Ela se comunica diretamente com o controlador. A camadas de Visão é a parte responsável pela interface do usuário. Nessa camada, os dados são requisitados ao Controlador que por sua vez requisitada ao modelo. O Controlador é a camada responsável intermediar a conexão entre o Modelo e a Visão.

2.3 Tecnologias e Ferramentas

As próximas subseções apresentam conceitos relacionados às tecnologias e às plataformas utilizadas para desenvolvimento do sistema.

2.3.1 Dart

Dart é uma linguagem desenvolvida pela Google, para criação de *softwares* multiplataforma. Seu lançamento ocorreu na GOTO Conference 2011, Dinamarca (RISSI; DALLILO, 2022).

Sua missão inicial era substituir o JavaScript como principal tecnologia presente nos navegadores, a ideia era oferecer uma alternativa mais moderna e robusta para o desenvolvimento *Web*. Um dos pontos que dificultaram sua adoção, no início, foi o JavaScript já ser uma linguagem consolidada no mercado, e também a má fama do Google em descontinuar projetos.

O Dart é uma linguagem orientada a objetos, mas pode ser considerada flexível, que permite que os programadores definam variáveis que aceitam todos os tipos de valores, chamadas de variáveis dinâmicas. A linguagem possui sintaxe com estilo baseado no C. Isso faz com que seja muito similar à linguagens como Java e C#.

2.3.2 Flutter

Flutter foi criada pelo Google como uma plataforma de desenvolvimento móvel de código aberto, também destinada a auxiliar os desenvolvedores na criação de interfaces para múltiplas plataformas. É construída na linguagem Dart, que é uma linguagem orientada a objetos que se compila em código nativo. É de fácil aprendizado para desenvolvedores provenientes de outras linguagens como Java e JavaScript, e oferece recursos como forte tipagem, genéricos e programação assíncrona (WU, 2018).

O Flutter é baseado no conceito de *widgets*, que são basicamente componentes *UI* que podem ser usados para criar interfaces de usuário complexas com apenas algumas linhas de código. Esse *framework* fornece um conjunto de *widgets* pré-construídos, mas também possibilita que os desenvolvedores façam seus próprios. Os *widgets* são classificados como *Stateless* e *Stateful widgets*, dependendo se eles gerenciam ou não algum estado. *Widgets Stateless* são aqueles que simplesmente exibem informações e não exigem nenhum dado mutável. Os *State-*

ful widgets, por outro lado, gerenciam o estado. Este estado pode ser interno ao próprio *widget*, ou pode vir de uma fonte externa. Para que um *Stateful Widget* possa mudar sua aparência em resposta a eventos, ele precisa ter acesso a um objeto imutável conhecido como uma instância *State* (WU, 2018; FENTAW, 2020).

Várias camadas compõem a estrutura Flutter (SHAH et al., 2019). A renderização da interface do usuário é de responsabilidade da camada *Widgets*. Ela usa uma abordagem declarativa para definir a interface de usuário. Isto significa que não é necessário escrever código imperativo para atualizar a interface de usuário. A camada *Engine*, por sua vez, está encarregada de coordenar a comunicação entre o *framework* e o código nativo da plataforma. Por fim, a camada de fundação fornece serviços fundamentais, incluindo rede, E/S e suporte de acessibilidade.

2.3.3 Python

Python é uma linguagem de programação de alto nível que se destaca por sua simplicidade e legibilidade. Criada pelo matemático Guido van Rossum, teve sua primeira versão lançada em 1990. Desde então, a linguagem tem evoluído com a colaboração de uma comunidade global de desenvolvedores e tornou-se uma das linguagens de programação mais populares do mundo (COUTO; SILVA, 2021).

O Python se caracteriza pela ênfase na legibilidade do código, o que o torna uma escolha ideal para programadores, desde iniciantes até profissionais experientes. A linguagem utiliza indentação para definir blocos de código, tornando o código mais claro e fácil de entender.

O Python se tornou uma parte importante da ciência de dados e da inteligência artificial. A ampla variedade de bibliotecas e *frameworks*, a facilidade na integração com outras tecnologias, a comunidade ativa, e o aprendizado de máquina e processamento de linguagem natural, são pontos que ajudam na adoção quando o assunto é inteligência artificial. Os cientistas e analistas utilizam códigos Python para analisar *big datas*, conjuntos de dados e projetar algoritmos de aprendizado de máquina (FOUNDATION, 2023).

2.3.4 Firebase

Firebase é uma plataforma de gerenciamento e desenvolvimento de dispositivos móveis, desenvolvida pela Firebase, Inc. em 2011 e adquirida pelo Google em 2014. A plataforma surgiu de uma necessidade dos desenvolvedores em gerenciar várias infraestruturas que faziam parte de um mesmo aplicativo. Com ela, os desenvolvedores podem focar na criação de sua própria aplicação, poupando tempo e recursos (LI et al., 2018).

Atualmente o Firebase é dividido em 3 partes nomeadas de : *Analytics*; *Develop* e *Grow*. O *Analytics* é a seção onde são disponibilizadas informações sobre o aplicativo em geral. É pos-

sível obter informações como: localidade, idade e sexo dos usuários, eventos que acontecem dentro do aplicativos e erros de sintaxe. O firebase oferece uma função de exportar todos os dados coletados para um *data warehouse* hospedado em nuvem, permitindo a criação de relatórios detalhados de dados de forma personalizada.

A seção de *Develop* é onde são localizadas as ferramentas de desenvolvimento que auxiliam o desenvolvedor. As informações são armazenadas no *Realtime Database*, que é uma base de dados hospedada em nuvens que utiliza a linguagem *NoSql*. Para manter a lógica e regra de negócio do *backend* centralizada e segura, foi criado o *Cloud Functions*, que exclui a necessidade de gerenciar o próprio servidor. Para a criação e gerencimento de domínios, existe o *Firebase Hosting*. Para *download* e *upload* de arquivos é utilizado o *Firebase Storage*. Os erros que acontecem nos aplicativos podem ser capturados pelo *Crash Reporting*, onde o desenvolvedor consegue a diagnosticar a causa das possíveis falhas. E por fim, opções de gerencimanento de desempenho e performance também são oferecidas pelo *Performance Monitoring* (CHATTERJEE et al., 2018).

A seções de *Grow* é a parte relacionada ao crescimento e engajamento. São fornecidas funções que ajudam na realização testes dentro do aplicativo, enviar mensagens para os usuário e visualizar padrões de comportamento dentro do aplicativo.

2.4 Classificação de alimentos e produtos

A classificação de produtos é feita com base nas 5 principais categorias de alimentos, de modo que cada alimento é colocado na categoria que tem seu macronutriente mais abundante, exceto frutas e vegetais. Sendo elas:

1. Fontes de Carboidratos
    (a) Grãos e cereais (pão, arroz, macarrão, aveia)
    (b) Leguminosas (feijões, lentilhas, grão-de-bico)
    (c) Vegetais ricos em amido (batatas, milho)
    (d) Produtos à base de amido (massas, tortilhas)
2. Fontes de Proteínas
    (a) Carnes magras (frango, peito de peru, carne magra)
    (b) Peixes (salmão, atum, tilápia)
    (c) Ovos
    (d) Tofu e outras alternativas à carne
3. Fontes de Gordura
    (a) Óleos (azeite de oliva, óleo de coco, óleo de canola)

   (b) Nozes e sementes (amêndoas, nozes, sementes de chia)

   (c) Abacate

   (d) Manteiga

4. Frutas

   (a) Maçãs

   (b) Bananas

   (c) Morango

   (d) Uva

5. Vegetais

   (a) Brócolis

   (b) Cenoura

   (c) Espinafre

   (d) Tomate

Além dessas categorias, existe também divisões por faixas de horário do dia. Desse modo, os alimentos são categorizados nas faixas de horário mais propícias para o consumo. Sendo essas faixas:

1. Manhã

   (a) 06:00 - 09:00

   (b) 09:00 - 12:00

2. Tarde

   (a) 12:00 - 15:00

   (b) 15:00 - 18:00

3. Noite

   (a) 18:00 - 21:00

   (b) 21:00 - 00:00

2.5 Trabalhos Relacionados

O primeiro artigo analisado é Tecnologia em Saúde: versão brasileira do *software* GloboDiet para avaliação do consumo alimentar em estudos epidemiológicos (STELUTI et al.,

2020). O *software* não apresenta análise de produtos, e a recomendação de produtos se baseia no consumo energético e nos macronutrientes dos alimentos.

O segundo artigo apresenta o CardNutri: Um *software* de planejamento de cardápios nutricionais semanais para alimentação escolar aplicando inteligência artificial (MOREIRA et al., 2017). A análise de produtos é feita por comparações de tabela de nutrientes, preço e localidade. A recomendação de alimentos é realizada por faixa etária, variedade de alimentos, harmonia de preparações, e um preço máximo a ser gasto para fazer cada refeição.

Como terceira análise realizada, foi utilizado um trabalho de conclusão de curso, que apresenta recomendação de presentes em dispositivos móveis. A análise do perfil de cada usuário é feita a partir de compras que foram realizadas, ou seja, conforme o histórico de produtos adquiridos é feito o mapeamento de interesses. Para a recomendação, os interesses são utilizados para recomendar produtos semelhantes aos que o usuário já adquiriu (PEREIRA; COSTA, 2016).

Este trabalho tem análise dos produtos baseada nas comparações das tabelas nutricionais, localidade do usuário, e o preço dos produtos, usando a localidade para aferir de forma mais precisa os preços. Além disso, a recomendação de alimentos é feita com base no plano alimentar, na localidade, no orçamento do paciente e nas preferências do usuário, que pode selecionar quais tipos de alimentos deseja consumir em quais faixas de horários do dia.

3 METODOLOGIA

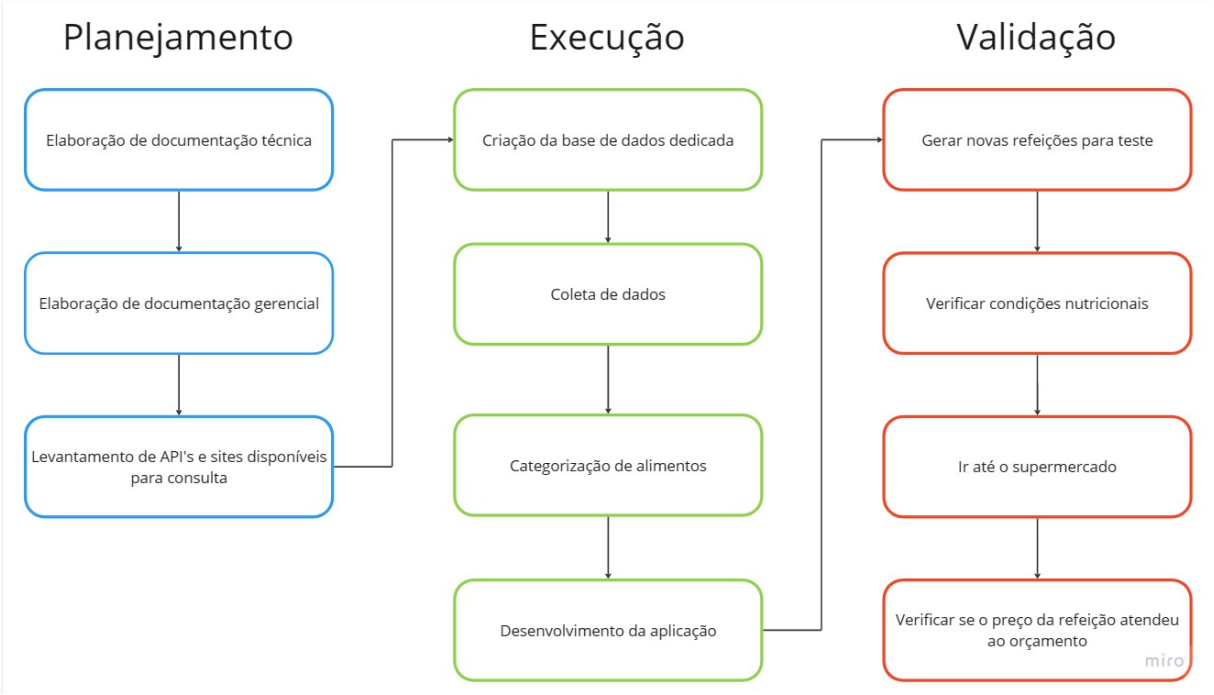
Para as, metodologias, foi adotado um modelo sequencial. Esse processo é controlado por um cronograma e é composto por três etapas: Planejamento, Execução e Validação, as quais são detalhadas na Figura 1.

3.1 Planejamento

Para o planejamento, foram definidos documentos técnicos em auxílio do processo de construção do *software*. Dentre eles estão:

- Levantamento de requisitos funcionais e não funcionais. São os nossos guias para a elaboração das funcionalidades do sistema. Tudo o que estiver contido nos documentos possibilita que o *software* seja atualizado e reparado sempre que necessário de acordo com o que foi inicialmente estipulado.
- Casos de uso baseados em histórias de usuários. É importante trazer situações reais para a aplicação, portanto a utilização de historias de usuários é indispensável.
- Restrição e mecanismos arquiteturais. Foi feito um levantamento das tecnologias a serem utilizadas para atender a todas as demandas funcionais e não funcionais do sistema.

Figura 1 – Etapas metodológicas



Fonte: elaborada pelos autores.

3.2 Execução

A etapa de execução foi subdividida entre: Elaboração de diagramas, Fonte de Dados e Desenvolvimento da aplicação. Os tópicos são exemplificados na Seção de Desenvolvimento.

3.2.1 Elaboração de diagramas

Ainda na área da documentação técnica, a elaboração de alguns diagramas é a primeira etapa da execução do projeto. Esses diagramas são facilitadores e fontes de consulta para o desenvolvimento da aplicação. Os documentos definidos para esta etapa foram:

- Documento arquitetural do sistema.
- Diagrama de classes.
- Diagrama de casos de uso.
- Requisitos funcionais e não-funcionais.

3.2.2 Fontes de dados

Este projeto utiliza duas fontes de dados principais para coletar informações nutricionais. Ambas as fontes são acessadas por meio de um *Web Crawler*, que é um programa automatizado que navega pela *web* e coleta informações. Essas duas fontes de dados, quando combinadas, fornecem uma visão completa e detalhada das informações nutricionais disponíveis, permitindo análises aprofundadas e precisas.

3.2.3 Desenvolvimento da aplicação

A etapa de desenvolvimento da aplicação foi subdividida entre: *back end* e *front end*.

3.2.3.1 Desenvolvimento do *back end*

O *back end* do projeto foi desenvolvido em Python, aproveitando as vantagens dessa linguagem de programação de alto nível, com uma comunidade robusta e vasta biblioteca de recursos para *Web Scraping* e *NoSQL*. Para a coleta de dados, utilizou-se a biblioteca *BeautifulSoup*, enquanto a criação de uma *API RESTful* foi realizada com a ajuda da biblioteca Flask. A implementação seguiu a arquitetura *RESTful*, promovendo a comunicação padronizada entre sistemas. O código foi organizado de maneira modular, simplificando tanto a manutenção quanto a expansão do sistema.

3.2.3.2 Desenvolvimento do *front end*

Para o desenvolvimento do *front end* foi utilizado o *framework* Flutter, ele é altamente eficiente na criação de aplicações móveis e *web* devido a várias razões-chave. Primeiramente, ele utiliza uma única base de código para desenvolver aplicativos tanto para dispositivos móveis (iOS e Android) quanto para a *web*, economizando tempo e recursos. Isso significa que os desenvolvedores podem criar e manter um único código-fonte em vez de lidar com duas implementações separadas. Para organização do código, foi adotada uma estrutura de módulos, deixando junto o controlador e a interface gráfica na mesma pasta. Essa arquitetura foi adotada pois em grandes sistemas é mais fácil dar manutenção em módulos separados.

3.3 Validação

Para esta última etapa, a validação foi feita na prática, pessoalmente, com a ida a alguns estabelecimentos. Após treinamento do sistema, alguns testes foram feitos. Os resultados desses

testes foram levados para os estabelecimentos que a aplicação abrange, colocando à prova se a aplicação realmente foi capaz de fazer um novo plano alimentar, com as mesmas condições nutricionais, mas com o preço final menor que a original, de modo a se adequar ao orçamento do paciente.

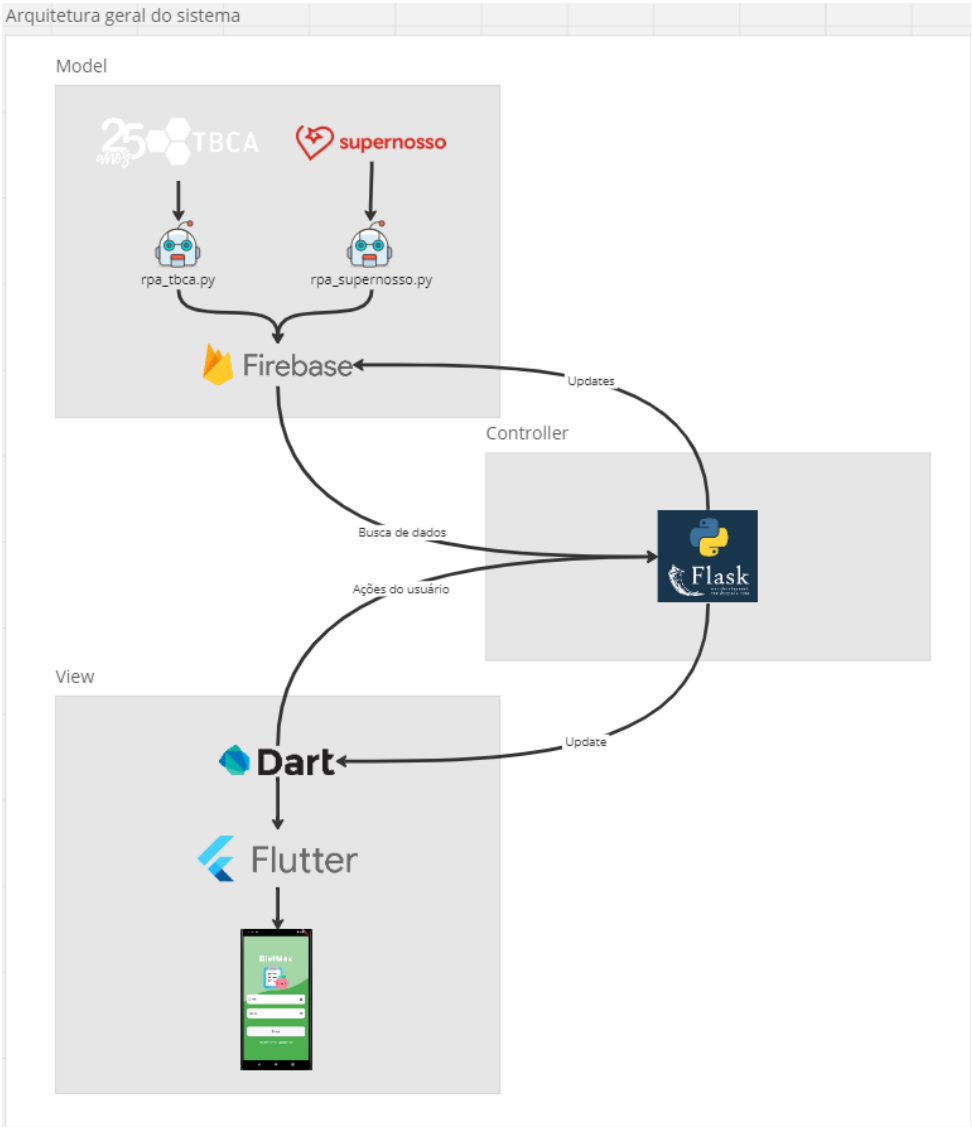
4 DESENVOLVIMENTO

Nesta seção, são apresentadas as três etapas do desenvolvimento da aplicação, das quais são: Planejamento, Execução e Validação.

4.1 Planejamento

Nessa seção, é mostrado como foi desenvolvido a aplicação. A Figura 2 apresenta a arquitetura geral do sistema, demonstrando o modelo MVC e as tecnologias utilizadas:

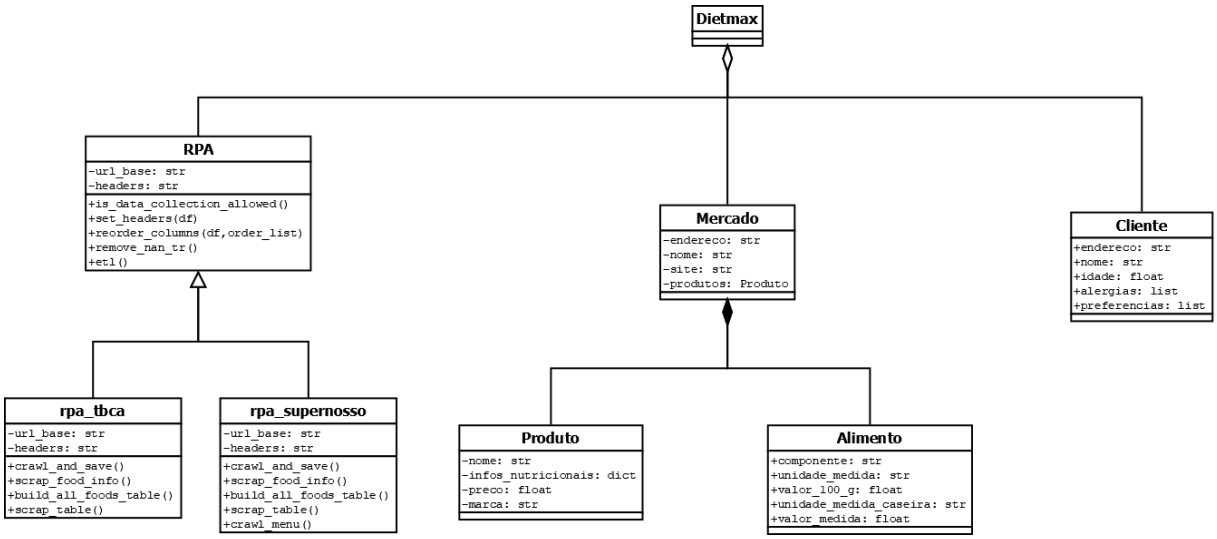
Figura 2 – Arquitetura geral do sistema



Fonte: elaborada pelos autores.

A Figura 3 representa o diagrama de classes do projeto, que é composto por 8 classes. Essas classes representam os *Web Crawlers*, os usuários do aplicativo "Clientes" e os alimentos que são sugeridos.

Figura 3 – Diagrama de classes



Fonte: elaborada pelos autores.

As Figuras 4 e 5, representam respectivamente, os requisitos funcionais e os requisitos não funcionais obtidos. Na primeira coluna da tabela são apresentados os requisitos, na segunda coluna, a descrição do requisito e, na terceira coluna, a classificação de prioridade.

Figura 4 – Requisitos Funcionais

Requisitos	Descrição	Prioridade
RF1: Inserir dieta	O aplicativo deve permitir que os usuários insiram informações sobre sua dieta e defina o horário de cada refeição.	Alta
RF2: Visualização da dieta	O aplicativo deve permitir a visualização simplificada da dieta inserida pelo usuário.	Alta
RF3: Visualizar valor nutricional	O aplicativo deve permitir que os usuários visualizem o valor nutricional dos alimentos.	Alta
RF4: Seleção de alimentos variados	O aplicativo deve fornecer ao usuário uma gama de alimentos diversificados para montar a sua dieta.	Média
RF5: Recomendação de alimentos	O aplicativo deve fornecer recomendações personalizadas com base nas informações inseridas pelo usuário	Alta
RF6: Edição dos dados do usuário	O aplicativo deve permitir o usuário alterar dados pessoais.	Baixa

Fonte: elaborada pelos autores.

Figura 5 – Requisitos Não-Funcionais

Requisitos	Descrição	Prioridade
RNF1: Usabilidade	O aplicativo deve ser de fácil utilização e ter uma interface amigável. Onde o usuário consiga se localizar sem muito esforço.	Média
RNF2: Desempenho	O aplicativo deve ser capaz de lidar com grandes quantidades de dados sem diminuir o desempenho.	Alta
RNF3: Experiência do Usuário	O aplicativo deve ser capaz de fornecer recomendações precisas e confiáveis com base nas informações inseridas pelo usuário.	Alta
RNF4: Segurança	O aplicativo deve ser seguro e proteger as informações do usuário.	Alta
RNF5: Portabilidade	O aplicativo deve ser compatível com diferentes dispositivos móveis. Sendo possível a utilização em Android e IOS.	Média

Fonte: elaborada pelos autores.

A Figura 6 representa as principais funcionalidades do sistema, fornecendo aos usuários a capacidade de criar planos alimentares personalizados, substituir alimentos, calcular custos, e receber recomendações de refeições adaptadas às suas preferências.

Figura 6 – Casos de uso

Fonte: elaborada pelos autores.

4.2 Execução

As telas do aplicativo, apresentada nesta subseção, foram projetadas para manter o fácil entendimento dos fluxos por qualquer faixa etária. Esta subseção se divide em: Fonte de dados, Desenvolvimento da aplicação e Validação.

4.2.1 Fontes de dados

Este trabalho utiliza duas fontes de dados principais para coletar informações nutricionais. Ambas as fontes são acessadas por meio de um *Web Crawler*, que é um programa automatizado que navega pela *web* e coleta informações. Essas duas fontes de dados, quando combinadas, fornecem uma visão completa e detalhada das informações nutricionais disponíveis,

permitindo análises aprofundadas e precisas.

4.2.1.1 Tabela Brasileira de Composição de Alimentos

A Tabela Brasileira de Composição de Alimentos (TBCA) é a primeira fonte de dados do projeto. A TBCA é uma compilação extensa e minuciosa de informações nutricionais relacionadas a uma vasta variedade de alimentos amplamente consumidos no Brasil. Essa valiosa fonte de dados é resultado de um estudo cuidadoso e abrangente que visa fornecer informações detalhadas sobre calorias, proteínas, carboidratos, gorduras, fibras, vitaminas e minerais presentes em cada alimento ((FORC), 2023).

4.2.1.2 *Site* do SuperNosso

O SuperNosso é um supermercado *online* que oferece uma variedade de produtos alimentícios, desde frutas e verduras até carnes e laticínios. Para obter as informações nutricionais desses produtos, o projeto utiliza um *Web Crawler*, que é um programa que navega pelo *site* do SuperNosso e extrai os dados relevantes. Dessa maneira, o projeto pode acessar uma grande quantidade de produtos e suas respectivas informações nutricionais, como calorias, proteínas, gorduras, vitaminas e minerais. Esses dados são essenciais para a análise e a recomendação de alimentos adequados ao orçamento e às preferências do usuário.

4.2.2 Desenvolvimento da aplicação

O desenvolvimento da aplicação é baseado em uma arquitetura MVC, sendo o componente *Model* feito em Python e Firebase, o componente *View* feito em Flutter e, por fim, o componente *Controller* feito em Python utilizando a biblioteca Flask. Essa escolha foi feita devido à facilidade que essas tecnologias nos proporcionam em cada um dos casos.

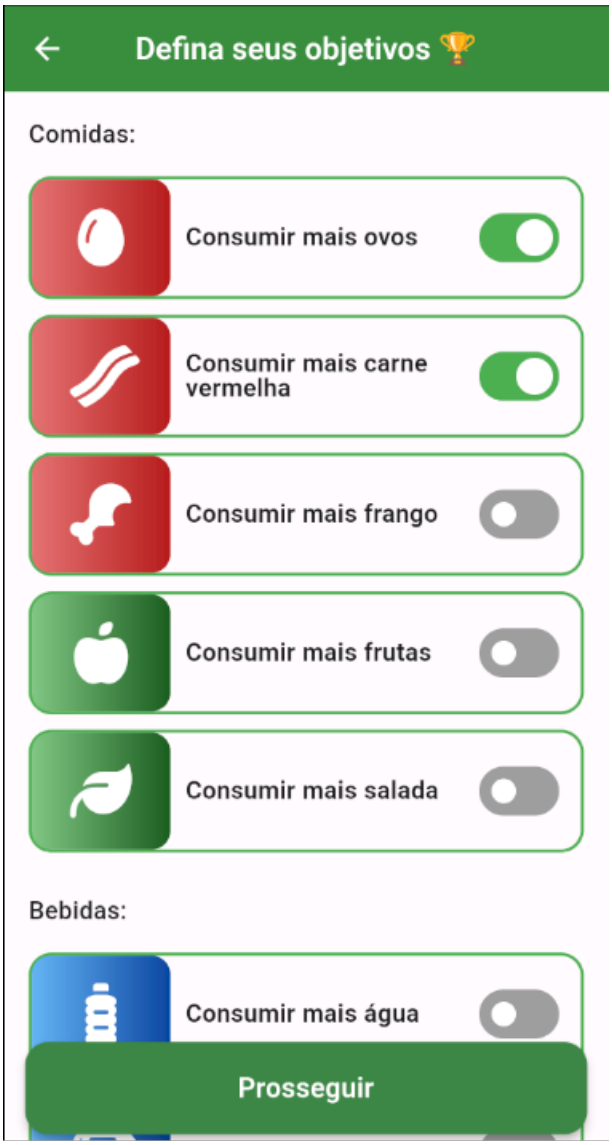
Model de persistência, inicialmente foi feito em Firebase, por ser simples de aplicar, manipular e ser capaz de lidar com a quantidade de dados estimada para uma versão beta. Além disso, para a coleta de dados foram criados *Web Crawlers* em Python que utilizam principalmente as bibliotecas *BeaultifulSoup, Requests e Pandas*.

View, foi escolhido o Flutter por ser uma linguagem eficaz, que gera aplicações nativas em diferentes Sistemas Operacionais e por ser uma linguagem bem documentada. Para a criação do *FrontEnd* da aplicação, foram utilizadas as bibliotecas *Get, Google Fonts, Font Awesome, Firebase e Dio*.

Controller foi feito em Python, utilizando a biblioteca Flask e essa escolha foi feita por ser a mesma linguagem do *Model*, o que facilita a comunicação entre os componentes. Foi criada uma *API* para comunicação com o componente *View*.

Para a criação da interface, foi utilizado a linguagem Dart e seu *framework* Flutter, dando assim a liberdade do sistema operar em multiplataformas. Para o desenvolvimento das telas, não foram encontrados grandes problemas. Para o *design* das telas foram realizadas buscas de inspirações em sistemas já existentes.

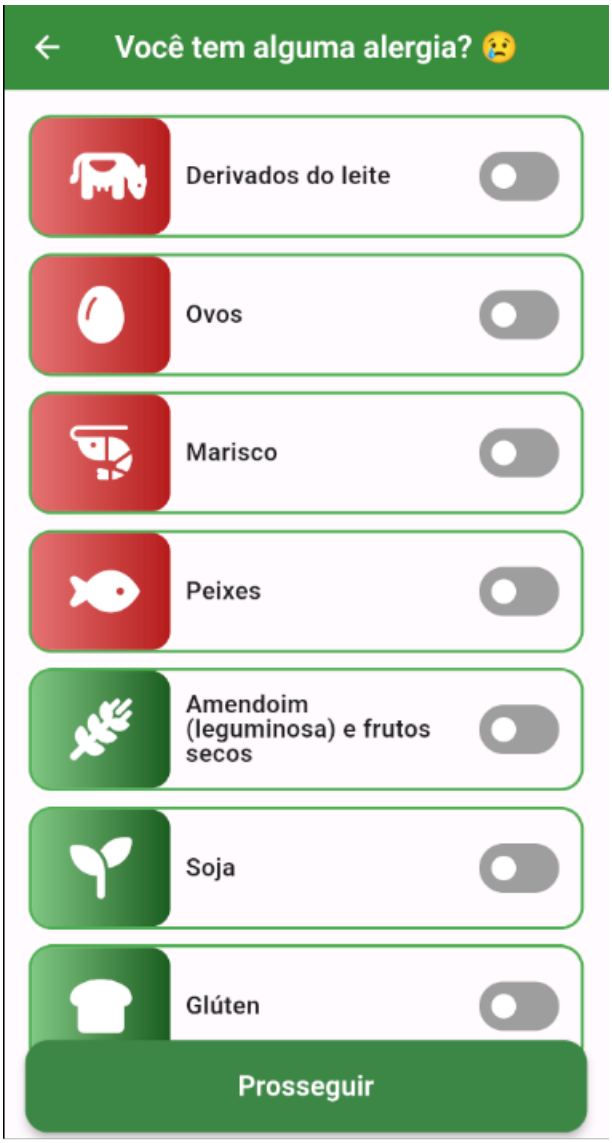
Figura 7 – Telas de Objetivos



Fonte: elaborada pelos autores.

Para iniciar o fluxo do aplicativo, é exibido a tela de objetivos, apresentada na Figura 7. Nesta tela, o usuário tem a possibilidade de selecionar alguns itens que podem ajudar a recomendação de alimentos.

Figura 8 – Telas de Preferências



Fonte: elaborada pelos autores.

Após clicar no botão de prosseguir da tela de objetivos, ilustrado na Figura 8, é exibido uma tela onde o usuário pode selecionar se possui algum tipo de alergia. Esta parte é muito importante para serem realizados filtros na recomendação de alimentos, removendo possíveis alimentos prejudiciais para o usuário.

Figura 9 – Telas de Dieta

Fonte: elaborada pelos autores.

Seguindo a tela das preferências, ilustrada na Figura 9, o usuário chega à parte onde pode ser visualizada toda a dieta adicionada. Nessa tela, é possível visualizar informações sobre quais alimentos foram adicionados na dieta, a quantidade, as calorias, e também é possível adicionar mais alimentos.

Figura 10 – Telas de Dieta

Fonte: elaborada pelos autores.

Ao clicar no botão de "Adicionar Alimentos", ilustrado na Figura 10, o usuário é enviado para a tela de seleção de alimentos. Nesta etapa é possível adicionar visualizar, e adicionar qualquer tipo de alimento dentro da dieta.

4.3 Validação

Para esta última etapa, a validação foi feita na prática, pessoalmente, com a ida ao Supernosso. Após treinamento do sistema, alguns testes foram feitos por pacientes reais para avaliar se a aplicação era fácil de usar, intuitiva e capaz de atender às suas necessidades específicas. Os resultados desses testes foram levados para o estabelecimento que a aplicação abrange, colocando à prova se a aplicação realmente era capaz de atender às necessidades dos usuários.

5 CONSIDERAÇÕES FINAIS

Referências

CHATTERJEE, Nilanjan et al. Real-time communication application based on android using google firebase. **IJARCSMS**, v. 6, 04 2018.

COUTO, Flávio M. do; SILVA, Julio F. da. Uso do módulo python uncertainties no cálculo de incertezas experimentais da diferença de potencial e corrente elétrica de um protótipo experimental. Campus de Alegre, Alegre, ES, BR, 2021. Disponvel em: https://www.scielo.br/j/rbef/a/zPhrQhWFVzGfcmVwmbkv8bs. Acesso em: 16 de out. 2023.

FENTAW, Awel Eshetu. Cross platform mobile application development: a comparison study of react native vs flutter. 2020.

(FORC), Universidade de São Paulo (USP). Food Research Center. **Tabela Brasileira de Composição de Alimentos (TBCA)**. 2023. <http://www.fcf.usp.br/tbca>. Acessado em: 28/09/2023.

FOUNDATION, Python Software. **Documentação Python**. [S.l.], 2023. Acesso em: 18 de out. 2023. Disponível em: <https://docs.python.org/pt-br/3/>.

LI, Wu-Jeng et al. Justiot internet of things based on the firebase real-time database. 2018. Disponível em: https://ieeexplore.ieee.org/abstract/document/8353979. Acesso em: 04 de set. 2023.

MAGNO, Fernanda Cristina Carvalho Mattos et al. **Macro e micronutrientes na orientação nutricional para obesidade**. 44. ed. Juiz de Fora, MG, BR: HU Revista, 2018. 251-259 p.

MOREIRA, Rafaela Priscila Cruz; MARTINS, Flávio Vinícius Cruzeiro; WANNER, Elizabeth Fialho. Cardnutri: Um software de planejamento de cardápios nutricionais semanais para alimentação escolar aplicando inteligência artificial. Rio de Janeiro, RJ, BR, 2017. Disponvel em: https://www.reciis.icict.fiocruz.br/index.php/reciis/article/view/1272. Acesso em: 24 de nov. 2022.

PEREIRA, Caíque de Paula; COSTA, Ruyther Parente da. **Algoritmo de Recomendação de Presentes em Dispositivos Móveis**. 2016 — Universidade de Brasília - UnB, acesso em: 18 de out. 2023. Disponível em: <https://fga.unb.br/articles/0001/6858/tcc-gifter-caique-ruyther.pdf>.

RISSI, Matheus; DALLILO, Felipe Diniz. Flutter um framework para desenvolvimento mobile. 2022. Disponvel em: https://recima21.com.br/index.php/recima21/article/view/2230. Acesso em: 04 de set. 2023.

SHAH, Kewal; SINHA, Harsh; MISHRA, Payal. Analysis of cross-platform mobile app development tools. In: IEEE. **2019 IEEE 5th International Conference for Convergence in Technology (I2CT)**. [S.l.], 2019. p. 1–7.

SILVA, Valéria Martins da. Revisão sistemática da evolução mvc na base acm. 2012. Disponível em: https://41jaiio.sadio.org.ar/sites/default/files/31$_E ST_2$012.$pdf. Acessoem: 04deset$.2023.

STELUTI, Josiane et al. Tecnologia em saúde: versão brasileira do software globodiet para avaliação do consumo alimentar em estudos epidemiológicos. São Paulo, SP, BR, 2020. Disponvel em: https://www.scielo.br/j/rbepid/a/DDF5kZKPwsbzBP3ZDDdTh8F/?lang=pt. Acesso em: 24 de nov. 2022.

WANGLER, Julian; JANSKY, Michael. The use of health apps in primary care–results from a survey amongst general practitioners in germany. **Wien Med Wochenschr**, v. 171, n. 7-8, p. 148–156, May 2021. Epub 2021 Feb 11.

WU, Wenhao. React native vs flutter, cross-platforms mobile application frameworks. Metropolia Ammattikorkeakoulu, 2018.