UNIVERSIDADE TUIUTI DO PARANÁ

ALISSON DAMBRATE CAMARGO

JOÃO VITOR TOKARSKI KAMINSKI

OTAVIO CALISTRO

**SOFTWARE DE RECOMENDAÇÃO DE PLANOS ALIMENTARES PERSONALIZADOS**

CURITIBA 2024

ALISSON DAMBRATE CAMARGO

JOÃO VITOR TOKARSK KAMINSKI

OTAVIO CALISTRO

**SOFTWARE DE RECOMENDAÇÃO DE PLANOS ALIMENTARES PERSONALIZADOS**

Trabalho conclusão de curso de Análise e Desenvolvimento de Sistemas da Universi- dade Tuiuti do Paraná.

Orientador: Angela Helena Zatti

Patrícia Rucker de Bassi

Sergio Luiz Marques Filho

CURITIBA 2024

**RESUMO.**

Este projeto visa desenvolver um sistema de recomendação de planos alimentares personalizados, utilizando as linguagens de programação Python, HTML, CSS e o framework Flask. A pesquisa aborda a necessidade de fornecer orientações nutricionais específicas baseadas nas características individuais dos usuários, como peso, altura, idade e nível de atividade física. O objetivo principal é criar um software que, após a coleta dessas informações, gere um plano alimentar detalhado e adequado às necessidades calóricas e nutricionais de cada usuário. O sistema será acessado por meio de um navegador web, permitindo que os usuários insiram seus dados de maneira simples e rápida.

A interface do usuário é projetada para ser extremamente intuitiva, facilitando a navegação e a utilização mesmo para aqueles que não possuem habilidades técnicas avançadas. A interface amigável, combinada com um design limpo e responsivo, garante uma experiência de usuário agradável e eficiente.

**Palavras- chave:**  Plano Alimentar, Software, Navegador Web, Dados, Experiência.

**LISTA DE FIGURAS.**

[FIGURA 1 – REQUISITOS FUNCIONAIS](#_bookmark2).....................................................................09

[FIGURA 2 – REQUISITOS NÃO FUNCIONAIS](#_bookmark3)............................................................09

[FIGURA 3 –](#_bookmark17) METODOLOGIA SCRUM.........................................................................13

[FIGURA 4 –](#_bookmark23) DIAGRAMAS DE CASO DE USO ..........................................................17

[FIGURA 5 –](#_bookmark23) CADASTRO E LOGIN...............................................................................24

[FIGURA 6 –](#_bookmark23) CÁLCULO TAXA METABOLICA BASAL (TBM) ....................................25

FIGURA 7 – AJUSTE DE CALORIAS COM BASE NA ATIVIDADE FÍSICA................26

FIGURA 8 – GERAÇÃO DE PLANO ALIMENTAR DIÁRIO.........................................27

FIGURA 9 – EXIBIÇÃO PALO ALIMENTAR.................................................................28

FIGURA 10 – DIAGRAMAS DE CLASSE USUÁRIO....................................................30

[FIGURA 11 – DIAGRAMAS DE CLASSE PLANO ALIMENTAR..................................31](#_Toc167389421)

[FIGURA 12 – DIAGRAMAS DE CLASSE REFEIÇÃO E ALIMENTO..........................31](#_Toc167389422)

[FIGURA 13 – CASO DE TESTE CADASTRAR USUÁRIO..........................................35](#_Toc167389423)

[FIGURA 14 – CASO DE TESTE EFETUAR LOGIN.....................................................36](#_Toc167389424)

[FIGURA 15 – CASO DE TESTE ENVIAR DADOS.......................................................37](#_Toc167389425)

[FIGURA 16 – CASO DE TESTE VISUALIZAR MAPA..................................................38](#_Toc167389426)

[FIGURA 17 – TELA INICIAL..........................................................................................42](#_Toc167389427)

[FIGURA 18 – TELA DE LOGIN.....................................................................................42](#_Toc167389428)

[FIGURA 19 – TELA DE ERRO......................................................................................43](#_Toc167389429)

[FIGURA 20 – TELA DE ERRO......................................................................................43](#_Toc167389430)

[FIGURA 21 – TELA DE DADOS....................................................................................44](#_Toc167389431)

[FIGURA 22 – TELA DE EXIBIÇÃO DE ALIMENTOS/HORÁRIOS..............................44](#_Toc167389432)

FIGURA 23 – MANUAL DO USUARIO.........................................................................46

FIGURA 24 –TELA DE LOGIN......................................................................................47

FIGURA 25 - DASHBOARD.........................................................................................48

FIGURA 26 – REFEIÇÕES DIARIAS............................................................................48

**SUMÁRIO.**

**1 INTRODUÇÃO**................................................................................................................7

**2 LEVANTAMENTO DE REQUISITOS E EXPLICAÇÃO DO CÁLCULO**.......................9

2.1 CÁLCULO DE TDEE..................................................................................................10

**3 METODOLOGIA DE CONSTRUÇÃO DA SOLUÇÃO**................................................10

3.1 APLOCAÇÃO DA METODOLOGIA...........................................................................11

3.2 TESTE PRÁTICO BASEADO NO NOSSO SOFTWARE.........................................11

3.3 REFERÊNCIA DO USO E POR QUE ESCOLHEMOS.............................................12

**4 FERRAMENTAS UTILIZADAS** ...................................................................................13

4.1 PYTHON.....................................................................................................................13

4.2 HTML (HYPERTEXT MARKUP LANGUAGE)..........................................................14

4.3 CSS (CASCADING STYLE SHEETS).......................................................................14

4.4 FLASK........................................................................................................................15

4.5 LISTAS.......................................................................................................................15

**5** **DIAGRAMAS** ...............................................................................................................15

5.1 DIAGRAMA DE CASO DE USO ...............................................................................16

5.1.1 EXPLICAÇÕES.......................................................................................................17

5.1.2 RELAÇÕES E INTERAÇÕES.................................................................................19

5.2 ESPECIFICAÇÃO DE CASO DE USO......................................................................19

5.3 DIAGRAMA DE SEQUÊNCIA....................................................................................22

5.3.1 IMPORTÂNCIA DO DIAGRAMA DE SEQUÊNCIA NO SISTEMA........................28

5.4 DIAGRAMA DE CLASSE ..........................................................................................29

5.4.1 DESCRIÇÃO DO DIAGRAMA DE CLASSES........................................................32

**6 RELÁTORIO**.................................................................................................................33

6.1 CASO DE TESTE...................................................................................................... 34

**7 SQLITE E LISTAS EM PYTHON**.................................................................................38

**8 TELAS**...........................................................................................................................42

**9** **CONCLUSÃO** ..............................................................................................................45

**10 MANUAL DO USUÁRIO** ............................................................................................42

10.1 CADASTRO DE USUÁRIO......................................................................................46

10.2 LOGIN......................................................................................................................47

10.3 DASHBOARD...........................................................................................................48

10.4 RECOMENDAÇÕES DE ALIMENTACÃO..............................................................49

11 REFERÊNCIAS...........................................................................................................50

**1 INTRODUÇÃO.**

Muitos usuários encontram dificuldade em obter orientações nutricionais que levem em consideração suas características pessoais, como peso, altura, idade e nível de atividade física. Sem essas informações, é desafiador seguir um plano alimentar que atenda às suas necessidades específicas. Este software coleta essas informações individuais e utiliza fórmulas de nutrição reconhecidas para calcular a Taxa Metabólica Basal (TMB) e a necessidade calórica diária total, ajustando o plano alimentar de acordo com o perfil de cada usuário. Além disso, planejar refeições balanceadas ao longo do dia pode ser complexo e demorado, especialmente para quem não possui conhecimento em nutrição. Isso pode levar a escolhas alimentares inadequadas ou a um desequilíbrio na ingestão de nutrientes. Para solucionar este problema, o sistema gera automaticamente um plano alimentar distribuído em seis refeições diárias, garantindo que o usuário receba recomendações claras e organizadas para cada refeição, promovendo uma alimentação balanceada e adequada.

O acesso a ferramentas nutricionais acaba sendo um pouco limitado, muitas das quais são complexas ou inacessíveis para usuários comuns. Algumas dessas ferramentas podem exigir conhecimentos técnicos ou um tempo de aprendizado significativo. Para superar esta barreira, o software é acessível via navegador web e possui uma interface intuitiva e fácil de usar. Isso permite que qualquer pessoa, independentemente de seu nível de conhecimento técnico, possa inserir suas informações e obter um plano alimentar personalizado de forma rápida e eficiente. Além disso, a dificuldade em seguir uma rotina alimentar adequada é uma preocupação comum. Muitas pessoas não conseguem manter a disciplina necessária para seguir uma dieta equilibrada devido à falta de estrutura e orientação. O software oferece uma solução estruturada, dividindo o plano alimentar em seis refeições diárias, o que ajuda os usuários a seguirem uma rotina consistente e saudável.

Outro problema enfrentado por muitas pessoas é a falta de conhecimento sobre como distribuir a ingestão calórica ao longo do dia. Alguns usuários podem consumir a maior parte de suas calorias em uma única refeição, o que pode levar a problemas como fadiga e má digestão. O software resolve esse problema fornecendo uma distribuição equilibrada das calorias ao longo do dia, assegurando que os usuários

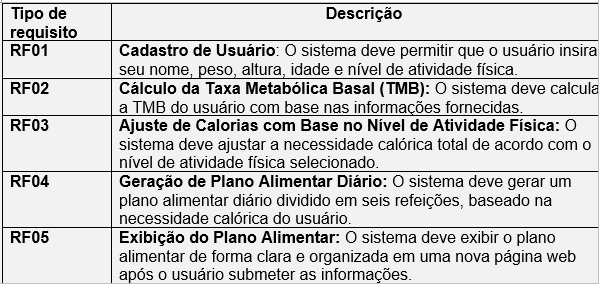
mantenham níveis de energia estáveis e uma digestão eficiente. Além disso, a variedade alimentar é um aspecto crítico para a saúde, e muitas pessoas tendem a

consumir os mesmos alimentos repetidamente por falta de conhecimento ou inspiração. O sistema pode sugerir uma variedade de alimentos para cada refeição, garantindo que os usuários recebam todos os nutrientes essenciais e mantenham uma dieta diversificada e interessante.

**2 LEVANTAMENTO DE REQUISITOS E EXPLICAÇÃO DO CÁLCULO.**

Os requisitos de um sistema de recomendação de planos alimentares personalizados são fundamentais para garantir que o software atenda às expectativas dos usuários e funcione de maneira eficiente.

FIGURA 1 – REQUISITOS FUNCIONAIS.



FONTE: O PRÓPRIO AUTOR.

FIGURA 2 – REQUISITOS NÃO FUNCIONAIS.

|  |  |
| --- | --- |
| **Tipo de requisito** | **Descrição** |
| **RNF01** | **Escalabilidade:** O sistema deve ser capaz de suportar um número crescente de usuários sem degradação significativa no desempenho. |
| **RNF02** | **Manutenibilidade:** O código do sistema deve ser escrito de forma clara e documentada, facilitando a manutenção e futuras atualizações. |
| **RNF03** | **Compatibilidade:** O sistema deve ser compatível com os principais navegadores web (Chrome, Firefox, Safari, Edge) e suas versões mais recentes. |
| **RNF04** | **Portabilidade:** O sistema deve ser facilmente portável para diferentes ambientes de hospedagem, permitindo migração entre servidores sem dificuldade significativa. |
| **RNF05** | **Reusabilidade:** Componentes de software devem ser desenvolvidos de forma a serem reutilizáveis em outros projetos ou partes do sistema. |

FONTE: O PRÓPRIO AUTOR.

2.1 CÁLCULO DE TDEE.

O Total Daily Energy Expenditure (TDEE) é a quantidade total de calorias que uma pessoa precisa consumir diariamente para manter seu peso, considerando seu nível de atividade física. O cálculo do TDEE dentro do sistema funciona da seguinte forma: Coleta de Dados: O usuário insere suas informações pessoais: nome, peso (kg), altura (cm), idade (anos) e nível de atividade física.

A TMB é calculada utilizando a fórmula de Harris-Benedict, que é diferente para homens e mulheres:

**Para homens:** TMB = 88.36 + (13.4 \* peso em kg) + (4.8 \* altura em cm) - (5.7 \* idade em anos)

**Para mulheres:** TMB = 447.6 + (9.2 \* peso em kg) + (3.1 \* altura em cm) - (4.3 \* idade em anos)

O nível de atividade física do usuário é utilizado para ajustar a TMB, multiplicando-a por um dos seguintes fatores:

**Sedentário (pouco ou nenhum exercício):** TMB \* 1.2

**Leve-Ativo (1-3 dias na semana):** TMB \* 1.375

**Moderadamente Ativo (3-5 dias na semana):** TMB \* 1.55

**Muito Ativo (6-7 dias na semana)**: TMB \* 1.725

**Super Ativo (exercícios intensos diários):** TMB \* 1.9

O resultado do ajuste é o TDEE, que representa a quantidade total de calorias que o usuário deve consumir diariamente para manter seu peso atual.

Com base no TDEE, o sistema distribui a necessidade calórica diária ao longo de seis refeições, garantindo uma ingestão balanceada e adequada de nutrientes.

**3 METODOLOGIA DE CONSTRUÇÃO DA SOLUÇÃO.**

No desenvolvimento do nosso sistema de planos alimentares personalizados, optamos por adotar a metodologia ágil. Dada a complexidade do projeto, acreditamos que essa abordagem é a mais adequada para lidar com as incertezas, as frequentes adaptações e a necessidade de entregas mais imediatas.

A metodologia ágil é especialmente adequada para projetos que envolvem uma grande quantidade de variáveis e potenciais mudanças nos requisitos. No caso do nosso sistema, vários fatores tornam a ágil a melhor escolha. A flexibilidade e adaptabilidade da

metodologia ágil permitem que nós respondamos rapidamente às mudanças nos requisitos nutricionais e de saúde dos usuários, que podem evoluir com base em novas descobertas científicas ou feedback dos usuários. Essa capacidade de ajustar o curso do projeto em resposta a novas informações ou feedback é crucial para garantir que o sistema continue a atender às necessidades dos usuários.

As vantagens da metodologia ágil no contexto deste projeto são múltiplas. Primeiramente, a capacidade de responder rapidamente a mudanças é essencial, especialmente quando se trata de um sistema que precisa se adaptar às necessidades e preferências em constante evolução dos usuários. A metodologia ágil permite que nossa equipe faça ajustes no produto conforme necessário, garantindo que ele permaneça relevante e útil.

3.1 APLICAÇÃO DA METODOLOGIA.

O desenvolvimento é dividido em sprints, que são ciclos de trabalho curtos. Cada sprint tem um conjunto claro de metas e termina com uma entrega funcional do software. Antes de cada sprint, há uma reunião de planejamento onde as tarefas do backlog (basicamente ítens que precisam ser desenvolvidos) são selecionadas e detalhadas. a equipe realiza reuniões rápidas (standups) para discutir o progresso durante todos os dias, identificar impedimentos e alinhar as atividades do dia. Estas reuniões são fundamentais para manter a comunicação fluida e resolver problemas rapidamente. No final de cada sprint, a equipe realiza uma revisão onde as funcionalidades desenvolvidas são testadas e avaliadas.

Esta revisão permite a coleta de um auto feedback, que é utilizado para ajustar o backlog e planejar os próximos sprints. Após cada sprint, a equipe realiza uma retrospectiva para discutir o que funcionou bem e o que pode ser melhorado. Esta prática de melhoria contínua é essencial para otimizar o processo de desenvolvimento e aumentar a eficiência.

3.2 TESTE PRÁTICO BASEADO NO NOSSO SOFTWARE.

**Primeira Iteração:**

* Objetivo: Implementar o cadastro de usuário e cálculo da TMB.
* Atividades: Desenvolvimento do formulário de entrada de dados, implementação das fórmulas de cálculo da TMB e testes iniciais.
* Entrega: Uma interface básica onde os usuários podem inserir seus dados e obter a TMB calculada.

**Segunda Iteração:**

* Objetivo: Adicionar o ajuste calórico baseado no nível de atividade física.
* Atividades: Integração do ajuste calórico com base nos níveis de atividade, testes de integração e validação de cálculos.
* Entrega: Uma versão do sistema que fornece a necessidade calórica diária total (TDEE).

**Terceira Iteração:**

* Objetivo: Geração e exibição do plano alimentar.
* Atividades: Desenvolvimento da lógica de distribuição de calorias em seis refeições, seleção de alimentos adequados e criação de uma interface de exibição.
* Entrega: Uma versão do sistema que exibe um plano alimentar detalhado para o usuário.

3.3 REFERÊNCIAS DO USO E POR QUE ESCOLHEMOS.

Pegamos alguns autores que reforçam o uso da metodologia ágil em nos projetos e decidimos citar alguns deles aqui como exemplo.

Segundo Schwaber e Sutherland (2020), criadores do Scrum, a metodologia ágil melhora a produtividade e a satisfação do cliente através de entregas frequentes e feedback contínuo. De acordo com o "Agile Manifesto" (Beck et al., 2001), os princípios ágeis são focados em indivíduos e interações, software funcional, colaboração com o cliente e resposta a mudanças, aspectos críticos para o sucesso do nosso projeto. Além disso, estudos como o de West e Grant (2010) indicam que equipes ágeis reportam uma redução significativa no tempo de desenvolvimento e uma melhora na qualidade do produto final, reforçando ainda mais nossa escolha pela metodologia ágil.

FIGURA 3 - METODOLOGIA SCRUM.



FONTE: https://www.euax.com.br/2022/10/metodologias-ageis

**4 FERRAMENTAS UTILIZADAS.**

4.1 PYTHON.

A linguagem Python desempenha um papel fundamental no desenvolvimento do nosso software, servindo como a linguagem de programação principal responsável por toda a lógica de negócios. Reconhecida por sua simplicidade, legibilidade e ampla gama de bibliotecas, Python oferece uma base sólida para construir nosso sistema de recomendação de planos alimentares personalizados.

O Python funciona como o motor do nosso software, interpretando e executando o código de forma eficiente e confiável. Sua sintaxe clara e estrutura intuitiva permitem uma implementação suave de todas as funcionalidades do sistema. Além disso, a extensa biblioteca padrão do Python fornece recursos robustos para lidar com diversas tarefas, desde o processamento de dados até cálculos complexos. O Python é utilizado para processar os dados inseridos pelos usuários, realizar cálculos de necessidades calóricas e nutricionais, selecionar alimentos adequados e gerar planos alimentares personalizados. Sua flexibilidade e poder computacional são essenciais para garantir que nosso sistema atenda às necessidades dos usuários de forma eficaz e eficiente.

4.2 HTML (Hypertext Markup Language)

O HTML é a espinha dorsal das nossas páginas web, fornecendo a estrutura básica para apresentação de conteúdo na internet. Como uma linguagem de marcação universalmente reconhecida, HTML desempenha um papel crucial na criação de uma experiência de usuário coesa e acessível.

Basicamente o HTML atua como o alicerce das nossas páginas web, definindo a estrutura hierárquica dos elementos de conteúdo. Seus elementos de marcação fornecem uma estrutura semântica para o conteúdo, permitindo que os navegadores interpretem e exibam as páginas de forma consistente. Isso inclui a criação de formulários para entrada de dados, exibição de resultados e organização de informações para fácil acesso pelos usuários. Seu foco é criar as interfaces de usuário, garantindo que os usuários possam interagir de forma eficiente com o sistema.

Ele define a estrutura das páginas, a disposição dos elementos e a hierarquia de informações, fornecendo uma base sólida para a apresentação de dados e funcionalidades.

4.3 CSS (Cascading Style Sheets).

Ele complementa o HTML, fornecendo estilos visuais para aprimorar a apresentação das nossas páginas web. Como uma linguagem de estilo poderosa, CSS permite que personalizemos a aparência e o layout das nossas páginas para criar uma experiência visualmente atraente e coesa.

O CSS funciona como a camada de estilo das nossas páginas web, controlando aspectos visuais como cores, fontes, espaçamento e layout. Suas regras de estilo são aplicadas aos elementos HTML, permitindo que customizemos a aparência de cada componente individualmente. Isso inclui a definição de cores de fundo, estilos de fonte, tamanhos de texto e posicionamento de elementos na página. No nosso software, CSS é utilizado para estilizar as páginas HTML, garantindo uma apresentação visualmente.

atraente e consistente. Ele define o estilo e o layout das páginas, criando uma experiência de usuário agradável e profissional.

4.4 Flask.

Flask é um framework web minimalista para Python, projetado para facilitar o desenvolvimento de aplicações web escaláveis e flexíveis. Como um componente central do nosso stack tecnológico, Flask oferece as ferramentas necessárias para criar um backend robusto e confiável para o nosso sistema. Funcionamento: Flask segue o padrão de arquitetura MVC (Model-View-Controller), onde as rotas (URLs) são mapeadas para funções Python que manipulam as requisições do usuário e retornam as respostas correspondentes.

Ele fornece um ambiente leve e flexível para o desenvolvimento web, permitindo que construamos uma aplicação web dinâmica e interativa de maneira eficiente. No nosso sistema, Flask é responsável por lidar com o lado do servidor da aplicação. Ele recebe as requisições dos usuários, executa a lógica de negócios em Python e retorna as páginas HTML resultantes. Com Flask, podemos criar um ambiente web dinâmico e interativo para os nossos usuários, proporcionando uma experiência de usuário suave e satisfatória.

4.5 LISTAS.

Ao utilizar listas em Python, podemos agilizar o processo de desenvolvimento, evitando a complexidade de configurar e gerenciar um banco de dados. Isso nos permite concentrar nossos esforços no desenvolvimento das funcionalidades principais do sistema, como o cálculo das necessidades nutricionais e a geração de planos alimentares personalizados. **No entanto, é importante ressaltar que essa abordagem é apenas para o estágio inicial do projeto.**

**5 DIAGRAMAS.**

Diagramas são representações visuais de informações ou processos que ajudam a entender e comunicar conceitos de forma mais clara e concisa. Eles são usados em diversas áreas, como engenharia, ciência da computação, arquitetura, entre outras, para descrever sistemas, estruturas, fluxos de trabalho e relações entre elementos.

* 1. DIAGRAMA DE CASO DE USO.

Um diagrama de caso de uso é um tipo específico de diagrama usado na engenharia de software para representar as interações entre os usuários de um sistema e o próprio sistema. Ele descreve as funcionalidades oferecidas pelo sistema do ponto de vista dos usuários, mostrando como os usuários interagem com o sistema para alcançar seus objetivos.

Em um diagrama de caso de uso, os elementos principais são:

* **Ator**: Representa um papel desempenhado por um usuário ou outro sistema externo que interage com o sistema sendo modelado. Pode ser uma pessoa, um dispositivo ou outro sistema.
* **Caso de Uso**: Representa uma funcionalidade ou conjunto de funcionalidades oferecidas pelo sistema. Cada caso de uso descreve uma interação específica entre o usuário e o sistema, geralmente focada em um objetivo particular do usuário.
* **Relacionamentos:** Os relacionamentos entre atores e casos de uso mostram como os atores estão envolvidos nas interações com o sistema. Por exemplo, um ator pode ser responsável por iniciar um caso de uso, ou um caso de uso pode envolver a interação de vários atores.

Esses diagramas ajudam a identificar os requisitos do sistema, a entender as necessidades dos usuários e a visualizar como o sistema será usado na prática.

FIGURA 4 – DIAGRAMAS DE CASO DE USO.

A diagram of a diagram

Description automatically generated

FONTE: O PRÓPRIO AUTOR.

* + 1. EXPLICAÇÕES.

**Ator**:

**Usuário**: Este é o único ator no diagrama, representando a pessoa que utiliza o sistema para obter um plano alimentar personalizado.

**Inserir Dados Pessoais:**

* **Descrição:** Este caso de uso envolve o usuário fornecendo suas informações pessoais, como nome, peso, altura e idade.
* **Interação:** O usuário interage com o sistema inserindo esses dados em um formulário.

**Selecionar Nível de Atividade Física:**

* **Descrição**: Neste caso de uso, o usuário seleciona seu nível de atividade física, que pode ser uma das seguintes opções: Sedentário, Leve-Ativo, Moderadamente Ativo, Muito Ativo ou Super Ativo.
* **Interação:** O usuário escolhe a opção que melhor descreve seu nível de atividade física através de uma interface de seleção.

**Enviar Dados:**

* **Descrição:** Depois de inserir todos os dados pessoais e selecionar o nível de atividade física, o usuário envia essas informações para o sistema para processamento.
* **Interação:** O usuário clica em um botão para submeter os dados inseridos.

Processos Internos (Incluídos no Enviar Dados):

**Calcular Necessidades Nutricionais:**

* **Descrição:** Este processo é responsável por calcular as necessidades calóricas e nutricionais do usuário com base nos dados fornecidos (peso, altura, idade e nível de atividade física).

**Relação:** Este caso de uso é incluído automaticamente quando o usuário envia osdados. É uma sub-ação do envio de dados.

**Gerar Plano Alimentar:**

* **Descrição:** Após calcular as necessidades nutricionais, o sistema gera um plano alimentar personalizado que inclui sugestões de alimentos para as diferentes refeições do dia (café da manhã, lanche, almoço, lanche da tarde, jantar e lanche noturno).
* **Relação:** Este também é um processo incluído automaticamente quando os dados são enviados.

**Exibir Plano Alimentar:**

* Descrição: O sistema exibe o plano alimentar gerado para o usuário, mostrando as sugestões de alimentos e as porções recomendadas para cada refeição.
* Relação: Este é o último passo incluído no envio de dados, apresentando o resultado final ao usuário.
  + 1. RELAÇÕES E INTERAÇÕES.
* **Usuário --> Inserir Dados Pessoais:** O usuário inicia o processo ao inserir suas

informações pessoais no sistema.

* **Usuário --> Selecionar Nível de Atividade Física:** O usuário continua o processo ao escolher seu nível de atividade física.
* **Usuário --> Enviar Dados:** O usuário finaliza a entrada de dados e submete as informações para processamento.
* **Enviar Dados --> Calcular Necessidades Nutricionais:** O sistema calcula automaticamente as necessidades nutricionais com base nos dados fornecidos.
* **Enviar Dados --> Gerar Plano Alimentar:** O sistema gera um plano alimentar personalizado com base nas necessidades nutricionais calculadas.
* **Enviar Dados --> Exibir Plano Alimentar:** Finalmente, o sistema exibe o plano alimentar gerado ao usuário.

5.2 ESPECIFICAÇÃO DE CASO DE USO.

Uma especificação de caso de uso é um documento que descreve detalhadamente como um sistema deve funcionar do ponto de vista do usuário. Ela fornece uma descrição completa de como os diferentes atores interagem com o sistema para atingir um objetivo específico. Cada caso de uso representa uma funcionalidade ou um conjunto de funcionalidades que o sistema oferece, e a especificação descreve o fluxo de eventos para essa funcionalidade, incluindo todas as possíveis interações, condições e exceções.

**Nome:**

Gerar Plano Alimentar

**Descrição:**

Este caso de uso descreve como o sistema gera um plano alimentar personalizado para o usuário com base nas suas necessidades calóricas e nutricionais, calculadas a partir dos dados fornecidos pelo usuário.

**Atores:**

Usuário

**Pré-condições:**

* O usuário deve ter inserido todos os dados pessoais necessários (nome, peso, altura, idade).
* O usuário deve ter selecionado o nível de atividade física.
* Os dados devem ter sido enviados ao sistema.

**Fluxo Principal:**

1. O caso de uso começa quando o sistema recebe os dados enviados pelo usuário.
2. O sistema calcula as necessidades calóricas e nutricionais do usuário com base nos dados fornecidos.
3. Com base nas necessidades calculadas, o sistema gera um plano alimentar personalizado que inclui sugestões de alimentos e porções para as seguintes refeições:

* Café da manhã
* Lanche da manhã
* Almoço
* Lanche da tarde
* Jantar
* Lanche noturno

1. O sistema organiza o plano alimentar de forma clara e acessível para o usuário.
2. O caso de uso termina quando o plano alimentar é exibido ao usuário.

**Fluxos Alternativos:**

**FA1: Dados Incompletos:**

* Se os dados fornecidos pelo usuário estiverem incompletos ou incorretos, o sistema exibe uma mensagem de erro solicitando a correção dos dados.
* O caso de uso volta ao ponto de entrada de dados, onde o usuário pode corrigir e reenviar as informações.

**FA2: Erro no Cálculo:**

* Se ocorrer um erro durante o cálculo das necessidades nutricionais, o sistema notifica o usuário do problema e solicita uma nova tentativa ou contato com o suporte técnico.
* O caso de uso termina com a exibição da mensagem de erro.

**Pós-condições:**

* O plano alimentar personalizado é exibido ao usuário.
* As necessidades calóricas e nutricionais foram corretamente calculadas e utilizadas na geração do plano.

**Requisitos Especiais:**

* O sistema deve ser capaz de processar rapidamente os cálculos nutricionais para evitar atrasos na geração do plano alimentar.
* A interface de exibição do plano alimentar deve ser clara, intuitiva e fácil de navegar.
* O sistema deve validar os dados de entrada para garantir que estão completos e corretos antes de iniciar o cálculo das necessidades nutricionais.

**Detalhamento do Processo:**

**Recepção dos Dados:** O sistema recebe os dados pessoais e o nível de atividade física do usuário através de um formulário submetido na interface web

**Cálculo das Necessidades Nutricionais:** Utilizando fórmulas específicas para cálculo do Total Daily Energy Expenditure (TDEE), o sistema calcula as necessidades calóricas e nutricionais do usuário. Isso leva em consideração fatores como idade, sexo, peso, altura e nível de atividade física.

1. **Geração do Plano Alimentar:** Baseado nas necessidades nutricionais calculadas, o sistema gera um plano alimentar que inclui sugestões de alimentos e porções específicas para as seis refeições do dia. As sugestões são selecionadas a partir de uma lista predefinida de alimentos saudáveis e balanceados.
2. **Exibição do Plano Alimentar:** O plano alimentar gerado é exibido ao usuário em uma interface clara e organizada. O usuário pode visualizar as sugestões de alimentos para cada refeição e as porções recomendadas.

5.3 DIAGRAMA DE SEQUÊNCIA.

Um diagrama de sequência é um tipo de diagrama UML (Unified Modeling Language) que mostra como os objetos interagem em um determinado processo, descrevendo a ordem em que essas interações ocorrem. Ele é particularmente útil para modelar a lógica temporal e o comportamento dinâmico de um sistema, detalhando o fluxo de mensagens trocadas entre os objetos para realizar uma tarefa específica.

Componentes do Diagrama de Sequência:

**Objetos:**

* Representados por retângulos no topo do diagrama, os objetos podem ser instâncias de classes, componentes ou atores que participam do processo. Cada objeto tem uma linha de vida (lifeline) que desce verticalmente a partir dele.

**Linha de Vida (Lifeline):**

* Uma linha pontilhada que desce do objeto e representa a existência do objeto ao longo do tempo. A linha de vida é utilizada para visualizar o tempo de vida do objeto durante a interação.

**Mensagens:**

* Representadas por setas horizontais entre as linhas de vida dos objetos, as mensagens indicam a comunicação entre os objetos. As mensagens podem ser síncronas (com uma ponta cheia) ou assíncronas (com uma ponta aberta).

**Ativação (Activation):**

* Retângulos finos sobre a linha de vida de um objeto que mostram quando e por quanto tempo um objeto está ativo ou executando um processo.

OBS: Ressaltando que usamos **FLASK** como servidor.

FIGURA 5 – CADASTRO E LOGIN.

**A diagram with text and images

Description automatically generated with medium confidence**

FONTE: O PRÓPRIO AUTOR,

**Descrição:**

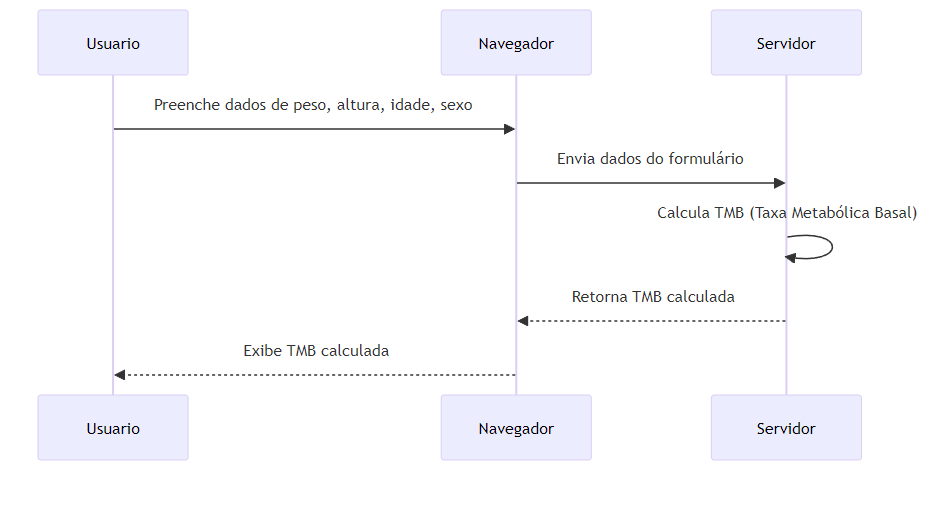
**Cadastro de Usuário**:

* **Usuário:** Abre o formulário de cadastro na interface do navegador.
* **Navegador:** Exibe o formulário de cadastro para o usuário.
* **Usuário**: Preenche o formulário de cadastro e envia os dados.
* **Navegador:** Envia os dados do formulário de cadastro para o servidor.
* **Servidor:** Recebe e processa os dados, salvando-os no banco de dados.
* **Servidor:** Retorna uma confirmação de cadastro para o navegador.
* **Navegador:** Exibe uma mensagem de sucesso de cadastro para o usuário.

**Login do Usuário:**

* **Usuário:** Abre o formulário de login na interface do navegador.
* **Navegador:** Exibe o formulário de login para o usuário.
* **Usuário:** Preenche o formulário de login e envia as credenciais.
* **Navegador:** Envia as credenciais do formulário de login para o servidor.
* **Servidor:** Verifica as credenciais no banco de dados.
* **Servidor:** Retorna uma confirmação de login para o navegador.
* **Navegador:** Exibe uma mensagem de sucesso de login para o usuário.

FIGURA 6 – CÁLCULO TAXA METABOLICA BASAL (TMB).

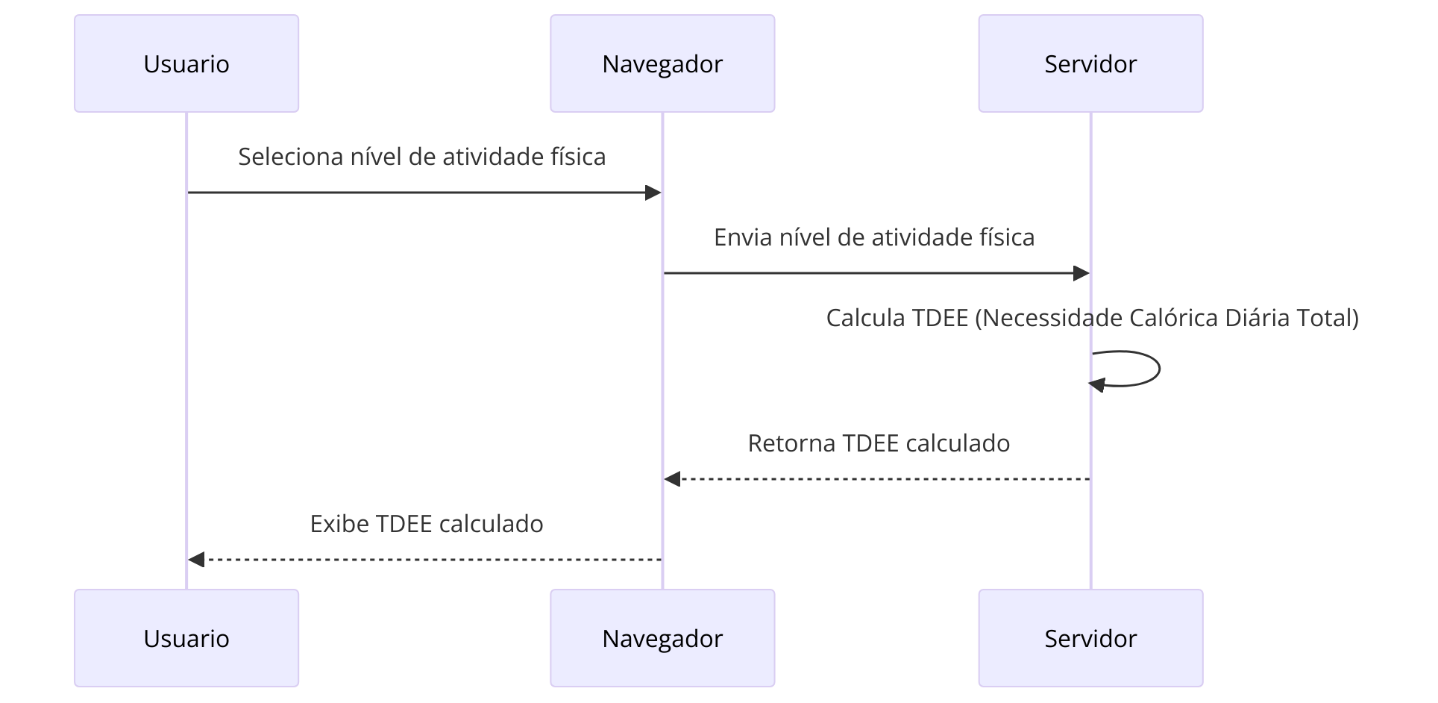


FONTE: O PRÓPRIO AUTOR.

**Descrição:**

* **Usuário:** Preenche dados de peso, altura, idade, e sexo no navegador.
* **Navegador:** Envia os dados do formulário para o servidor.
* **Servidor:** Calcula a TMB com base nos dados recebidos.
* **Servidor:** Retorna a TMB calculada para o navegador.
* **Navegador:** Exibe a TMB calculada para o usuário.

FIGURA 7 – AJUSTE DE CALORIAS COM BASE NA ATIVIDADE FÍSICA

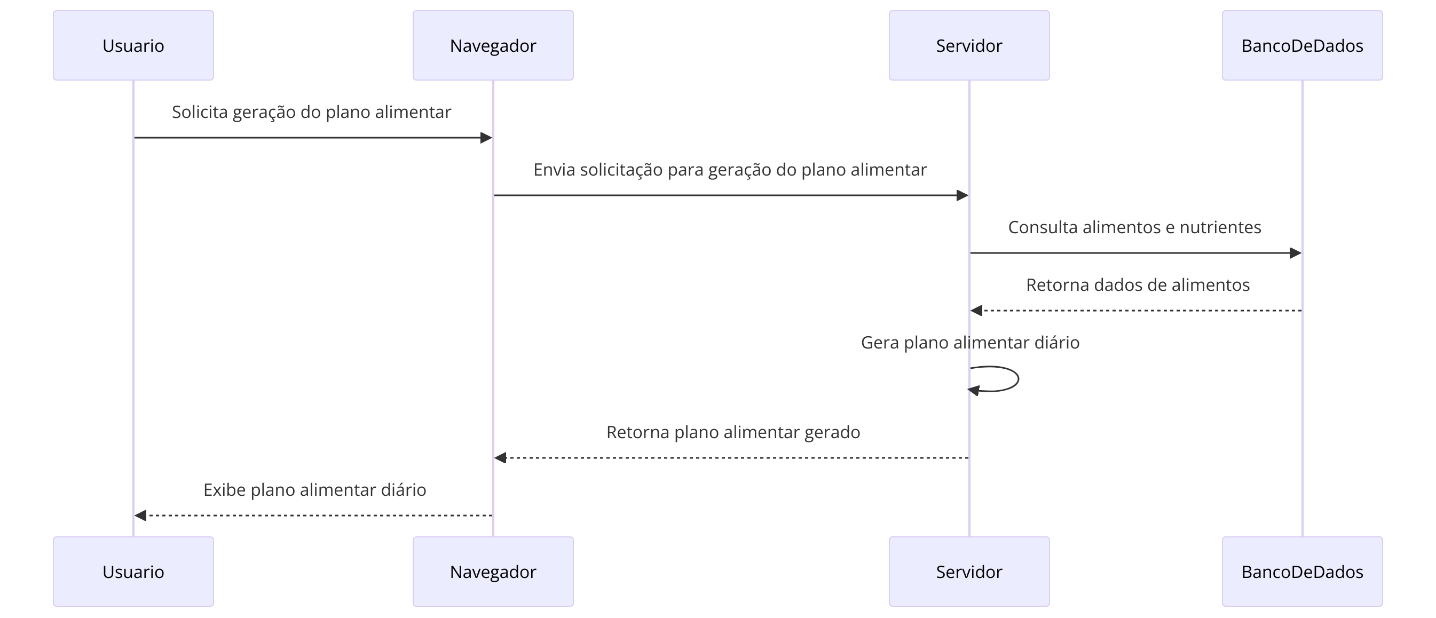


FONTE: O PRÓPRIO AUTOR.

**Descrição:**

* **Usuário:** Seleciona o nível de atividade física no navegador.
* **Navegador:** Envia o nível de atividade física para o servidor.
* **Servidor:** Calcula o TDEE (Necessidade Calórica Diária Total) ajustando a TMB com base no nível de atividade física.
* **Servidor:** Retorna o TDEE calculado para o navegador.
* **Navegador:** Exibe o TDEE calculado para o usuário.

FIGURA 8 – GERAÇÃO DE PLANO ALIMENTAR DIÁRIO.

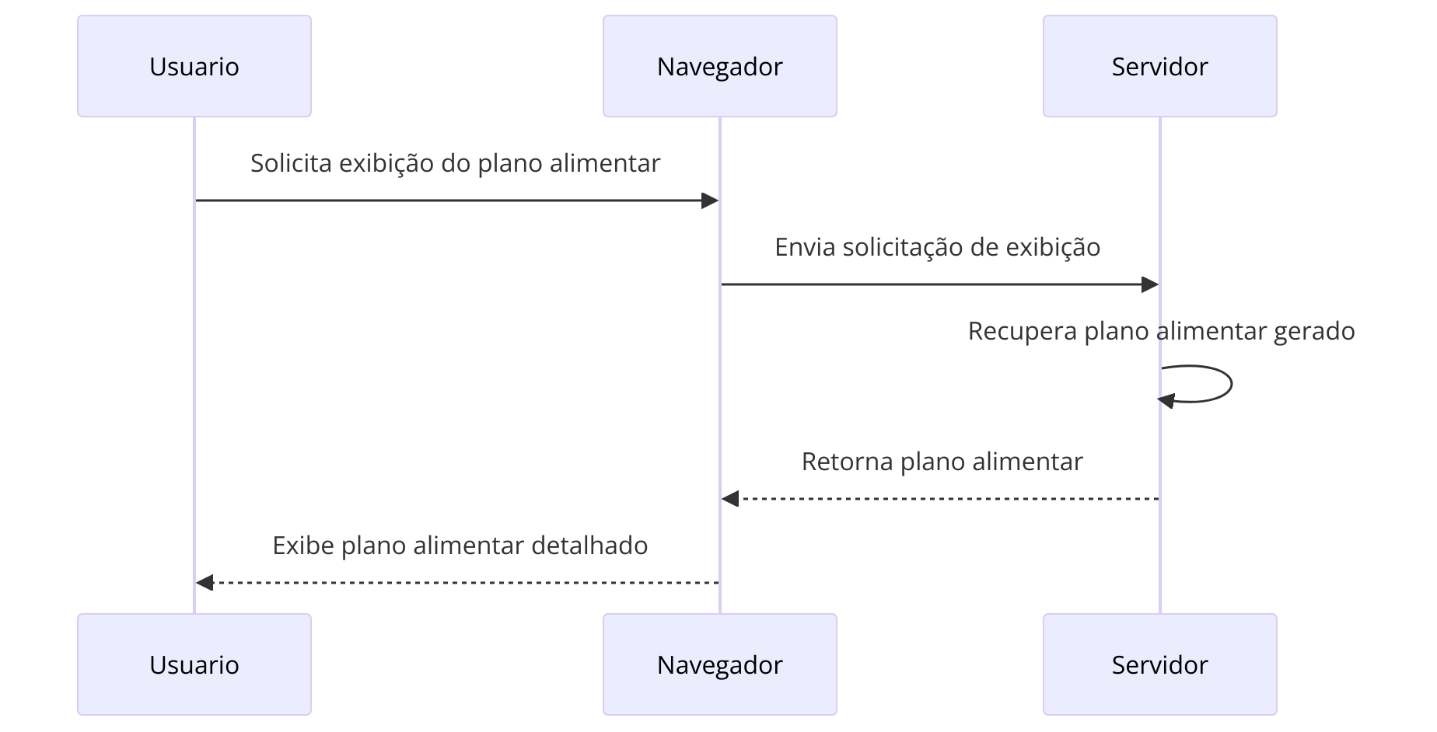


FONTE: O PRÓPRIO AUTOR.

**Descrição:**

* **Usuário:** Solicita a geração do plano alimentar no navegador.
* **Navegador:** Envia a solicitação para o servidor.
* **Servidor**: Consulta o banco de dados para obter informações sobre alimentos e nutrientes.
* **Banco de Dados:** Retorna os dados dos alimentos para o servidor.
* **Servidor:** Gera o plano alimentar diário com base nas necessidades calóricas do usuário.
* **Servidor:** Retorna o plano alimentar gerado para o navegador.
* **Navegador:** Exibe o plano alimentar diário para o usuário.

FIGURA 9 – EXIBIÇÃO PLANO ALIMENTAR..



FONTE: O PRÓPRIO AUTOR.

**Descrição:**

* **Usuário:** Solicita a exibição do plano alimentar no navegador.
* **Navegador:** Envia a solicitação de exibição para o servidor.
* **Servidor:** Recupera o plano alimentar gerado anteriormente.
* **Servidor:** Retorna o plano alimentar para o navegador.
* **Navegador:** Exibe o plano alimentar detalhado para o usuário.

5.3.1 IMPORTÂNCIA DO DIAGRAMA DE SEQUÊNCIA NO SISTEMA.

Para o nosso sistema de recomendação de planos alimentares personalizados, os diagramas de sequência ajudam a:

* Visualizar o processo de cadastro e cálculo das necessidades nutricionais.
* Entender como o sistema gera e exibe o plano alimentar diário.
* Planejar e comunicar de forma eficiente as interações entre o frontend (navegador) e o backend (servidor e banco de dados).
* Garantir que todas as etapas do processo sejam consideradas e implementadas corretamente, desde a entrada de dados pelo usuário até a exibição dos resultados.

5.4 DIAGRAMA DE CLASSE.

Um diagrama de classes é um tipo de diagrama estático na modelagem de sistemas orientados a objetos. Ele é parte da Unified Modeling Language (UML) e é amplamente utilizado para descrever a estrutura de um sistema mostrando suas classes, atributos, métodos e os relacionamentos entre os objetos. Aqui estão alguns pontos-chave sobre diagramas de classes:

Classes.

* **Definição:** Representam os principais componentes de um sistema. Cada classe é uma abstração de um objeto do mundo real ou conceitual, com propriedades e comportamentos.
* **Componentes da Classe:** Uma classe é geralmente representada por um retângulo dividido em três partes:

1. **Nome da Classe:** Na parte superior, que indica o nome da classe.
2. **Atributos:** Na parte do meio, que lista as propriedades ou dados que a classe armazena.

3. **Métodos:** Na parte inferior, que lista as funções ou operações que a classe pode realizar.

Atributos.

* **Definição:** Propriedades ou características que os objetos de uma classe possuem. Cada atributo tem um nome e um tipo de dado.
* **Visibilidade:** Podem ter diferentes níveis de visibilidade (público, privado, protegido), indicados por símbolos como + (público), - (privado) e # (protegido).

Métodos.

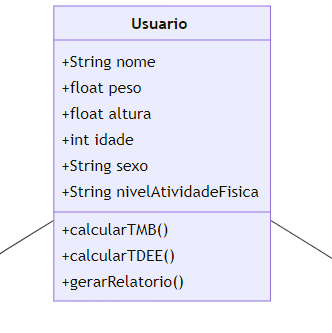
* **Definição:** Funções ou operações que podem ser executadas pelos objetos de uma classe.
* **Visibilidade:** Similar aos atributos, podem ser públicos, privados ou protegidos.

Relacionamentos.

* **Associação:** Indica um relacionamento entre duas classes, onde uma classe "usa" ou "refere-se" à outra.
* **Unidirecional:** Apenas uma classe sabe sobre a existência da outra.

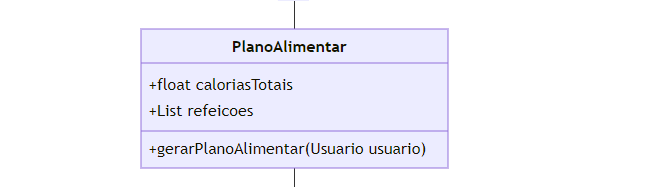
* **Bidirecional:** Ambas as classes sabem sobre a existência uma da outra.
* **Agregação:** Um tipo especial de associação que indica um relacionamento "todo/parte" onde uma classe é composta de outras classes. Representada por um losango aberto na extremidade da classe "todo".
* **Composição:** Um tipo mais forte de agregação onde as partes não podem existir sem o todo. Representada por um losango preenchido na extremidade da classe "todo".
* **Herança (Generalização):** Indica que uma classe (subclasse) herda atributos e métodos de outra classe (superclasse). Representada por uma linha com um triângulo vazio na extremidade da superclasse.
* **Dependência:** Um relacionamento fraco que indica que uma classe depende da outra para funcionar. Representada por uma linha tracejada com uma seta.

# FIGURA 10 – DIAGRAMAS DE CLASSE USUÁRIO.



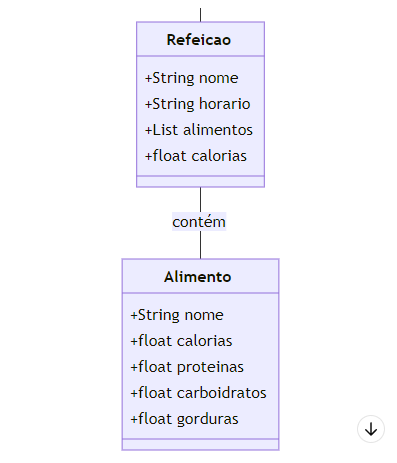
FONTE: O PRÓPRIO AUTOR.

# FIGURA 11 – DIAGRAMAS DE CLASSE PLANO ALIMENTAR.



FONTE: O PRÓPRIO AUTOR.

# FIGURA 12 – DIAGRAMAS DE CLASSE REFEIÇÃO E ALIMENTO.



FONTE: O PRÓPRIO AUTOR.

5.4.1DESCRIÇÃO DO DIAGRAMA DE CLASSES.

Classe Usuario.

**Atributos:**

* nome: String
* peso: float
* altura: float
* idade: int
* sexo: String
* nivelAtividadeFisica: String

**Métodos:**

* calcularTMB()
* calcularTDEE()

Classe Plano Alimentar.

**Atributos:**

* caloriasTotais: float
* refeicoes: List<Refeicao>

**Métodos:**

* gerarPlanoAlimentar(usuario: Usuario)

Classe Refeicao.

**Atributos:**

* nome: String
* horario: String
* alimentos: List<Alimento>
* calorias: float

Classe Alimento.

**Atributos:**

* nome: String
* calorias: float
* proteinas: float
* carboidratos: float
* gorduras: float

Relacionamentos entre Classes:

**Usuário** gera um **PlanoAlimentar** (Usuario -- PlanoAlimentar : gera)

**PlanoAlimentar** contém **várias Refeicao** (PlanoAlimentar -- Refeicao : contém)

**Refeicao** contém vários **Alimento** (Refeicao -- Alimento : contém)

**6 RELÁTORIO.**

O objetivo deste relatório é garantir que o Sistema de Recomendação de Planos Alimentares Personalizados atenda aos requisitos funcionais e de desempenho, proporcionando uma experiência confiável e segura aos usuários. O plano de testes inclui testes de unidade, testes de integração, testes de sistema e testes de aceitação.

**Plano de teste.**

Testes de Unidade.

* **Validação de Usuário:** Verificar a correta validação dos campos de entrada do usuário, como nome, peso, altura, idade, nível de atividade física e e-mail.
* **Cálculo da TMB:** Garantir que o sistema calcula corretamente a Taxa Metabólica Basal (TMB) com base nas informações fornecidas pelo usuário.
* **Geração de Plano Alimentar:** Verificar a geração correta dos planos alimentares diários com base nas necessidades calóricas do usuário.

Testes de Integração.

* **Registro de Usuário:** Verificar o armazenamento correto das informações do usuário no banco de dados após o registro.
* **Login de Usuário:** Garantir que o sistema permita que o usuário faça login com sucesso após o registro.
* **Geração e Exibição do Plano Alimentar:** Verificar a integração entre o cálculo das necessidades nutricionais, a geração do plano alimentar e a exibição dos resultados ao usuário.

Testes de Sistema.

* **Segurança de Dados:** Garantir que os dados do usuário sejam armazenados de forma segura e protegidos contra ameaças de segurança.
* **Usabilidade:** Avaliar a interface do usuário para garantir que seja amigável e fácil de usar.
* **Escalabilidade:** Avaliar o desempenho do sistema sob carga, garantindo que possa lidar com um grande número de solicitações simultâneas.

**Cronograma de Testes.**

Os testes serão realizados em paralelo com o desenvolvimento do sistema. Cada fase de desenvolvimento será seguida por testes correspondentes para garantir a qualidade contínua. O cronograma específico será definido de acordo com a evolução do projeto, priorizando áreas críticas e funcionalidades principais.

Responsabilidades

* **Equipe de Desenvolvimento:** Responsável pela realização dos testes de unidade e integração.
* **Testadores Externos (Usuários Reais):** Recrutados para os testes de aceitação, proporcionando feedback valioso baseado em experiências reais.
* **Equipe de Desenvolvimento e Testadores Externos:** Compartilharão responsabilidades nos testes de sistema, colaborando para garantir uma cobertura completa e identificar quaisquer áreas de melhoria.

* 1. CASOS DE TESTE.

Casos de teste são descrições detalhadas de cenários específicos usados para verificar se um sistema, aplicativo ou uma parte de um software funciona corretamente. Eles são projetados para garantir que todos os requisitos especificados para o software estejam sendo atendidos e que o sistema esteja funcionando conforme esperado.

# FIGURA 13 – CASO DE TESTE CADASTRAR USUÁRIO.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID do Teste** | **Teste** | **Descrição** | **Entrada** | **Resultado Esperado** |
| CT01 | Validação de Campos Vazios | Verificar se o sistema valida campos obrigatórios não preenchidos | Nome vazio, senha vazia | Mensagens de erro indicando que os campos são obrigatórios |
| CT02 | Validação de Nome Curto | Verificar se o sistema valida o nome com menos de 2 caracteres | Nome: "A", Senha: "senha123" | Mensagem de erro indicando que o nome deve ter mais de um caractere |
| CT03 | Validação de Senha Curta | Verificar se o sistema aceita uma senha com pelo menos 1 caractere | Nome: "João", Senha: "1" | Cadastro aceito, dados armazenados corretamente no banco de dados |
| CT04 | Cadastro com Dados Válidos | Verificar se o sistema aceita e armazena dados válidos | Nome: "João", Senha: "senha123" | Cadastro aceito, dados armazenados corretamente no banco de dados |
| CT05 | Persistência de Dados | Verificar se os dados do usuário são persistidos corretamente no banco de dados | Nome: "João", Senha: "senha123" | Dados armazenados corretamente no banco de dados |

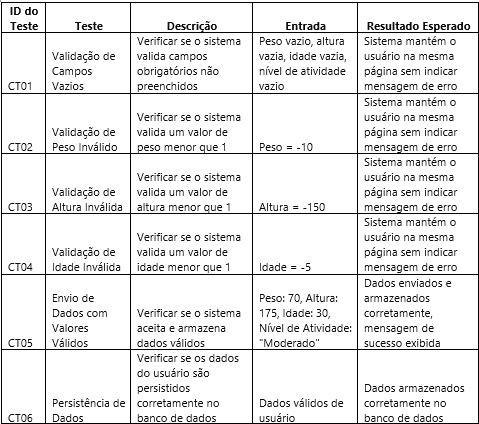
FONTE: O PRÓPRIO AUTOR.

# FIGURA 14 – CASO DE TESTE EFETUAR LOGIN.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID do Teste** | **Teste** | **Descrição** | **Entrada** | **Resultado Esperado** |
| A | Validação de Campos Vazios | Verificar se o sistema valida campos obrigatórios não preenchidos | Nome vazio, senha vazia | Mensagens de erro indicando que os campos são obrigatórios |
| B | Validação de Senha Curta | Verificar se o sistema valida a senha com menos de 2 caracteres | Nome: "João", Senha: "1" | Mensagem de erro indicando que a senha deve ter pelo menos 2 caracteres |
| C | Validação de Senha Incorreta | Verificar se o sistema não permite login com senha incorreta | Nome: "João", Senha: "senhaErrada" | Mensagem de erro indicando que a senha está incorreta |
| D | Validação de Nome Não Cadastrado | Verificar se o sistema não permite login com nome não cadastrado | Nome: "Maria", Senha: "senha123" | Mensagem de erro indicando que o nome não está cadastrado |
| E | Login com Dados Válidos | Verificar se o sistema permite login com nome e senha válidos | Nome: "João", Senha: "senha123" | Login bem-sucedido, redirecionamento para a página inicial |
| F | Manutenção de Sessão | Verificar se o sistema mantém a sessão do usuário após o login | Login com dados válidos | Sessão mantida enquanto o usuário estiver ativo |

FONTE: O PRÓPRIO AUTOR.

# FIGURA 15 – CASO DE TESTE ENVIAR DADOS.



FONTE: O PRÓPRIO AUTOR.

# FIGURA 16 – CASO DE TESTE VISUALIZAR MAPA.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID do Teste** | **Teste** | **Descrição** | **Entrada** | **Resultado Esperado** |
| CT01 | Validação de Exibição Correta | Verificar se o plano alimentar é exibido corretamente com todas as refeições | Dados válidos do usuário e cálculo de necessidades nutricionais | Plano alimentar exibido corretamente com todas as refeições detalhadas |
| CT02 | Exibição de Mensagem de Sucesso | Verificar se o sistema exibe uma mensagem de sucesso ao exibir o plano alimentar | Dados válidos do usuário e cálculo de necessidades nutricionais | Mensagem de sucesso exibida junto com o plano alimentar |
| CT03 | Persistência de Exibição de Dados | Verificar se os dados do plano alimentar permanecem exibidos após a atualização da página | Dados válidos do usuário e cálculo de necessidades nutricionais | Dados do plano alimentar continuam exibidos corretamente após atualização da página |
| CT04 | Exibição com Dados Incompletos | Verificar o comportamento do sistema quando dados incompletos são fornecidos para o cálculo do plano alimentar | Dados incompletos (ex: falta de peso ou altura) | Sistema mantém o usuário na mesma página sem indicar mensagem de erro |

FONTE: O PRÓPRIO AUTOR.

**7 SQLITE E LISTAS EM PYTHON.**

A combinação de SQLite e listas em Python permite que nosso sistema aproveite o melhor de ambos os mundos. Utilizamos SQLite para garantir que os dados do cadastro dos usuários sejam armazenados de forma segura e organizada, enquanto as listas em Python proporcionam a flexibilidade e a rapidez necessárias para gerar e manipular as recomendações de alimentação. Essa abordagem híbrida facilita tanto o desenvolvimento quanto a manutenção contínua do software, garantindo que possamos atender às diferentes necessidades do nosso sistema de maneira eficiente e eficaz.

O armazenamento dos dados ocorre nas seguintes partes do código:

1. Configuração do Banco de Dados.

O banco de dados é configurado em app.py:

app.config['SQLALCHEMY\_DATABASE\_URI'] = 'sqlite:///site.db'

db = SQLAlchemy(app)

2. Definição dos Modelos.

Os modelos User e UserInfo são definidos em models.py:

Python.

Copiar código

class User(db.Model):

id = db.Column(db.Integer, primary\_key=True)

username = db.Column(db.String(20), unique=True, nullable=False)

email = db.Column(db.String(120), unique=True, nullable=False)

password = db.Column(db.String(60), nullable=False)

user\_info = db.relationship('UserInfo', backref='user', lazy=True)

class UserInfo(db.Model):

id = db.Column(db.Integer, primary\_key=True)

name = db.Column(db.String(100), nullable=False)

weight = db.Column(db.Float, nullable=False)

height = db.Column(db.Float, nullable=False)

age = db.Column(db.Integer, nullable=False)

gender = db.Column(db.String(10), nullable=False)

physical\_activity = db.Column(db.String(50), nullable=False)

user\_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=False)

date\_created = db.Column(db.DateTime, nullable=False, default=datetime.utcnow)

3. Registro do Usuário.

Quando um usuário se registra, as informações são salvas no banco de dados na rota /register:

@app.route('/register', methods=['GET', 'POST'])

def register():

form = RegistrationForm()

if form.validate\_on\_submit():

user = User(username=form.username.data, email=form.email.data, password=form.password.data)

db.session.add(user)

db.session.commit()

flash('Registrado com sucesso! Faça o login para acessar sua conta.', 'success')

return redirect(url\_for('login'))

return render\_template('register.html', title='Registro', form=form)

4. Informações do Usuário.

Quando um usuário preenche o formulário de informações pessoais, os dados são salvos no banco de dados na rota /userinfo:

python

Copiar código

@app.route('/userinfo', methods=['GET', 'POST'])

def userinfo():

form = UserInfoForm()

if form.validate\_on\_submit():

user\_info = UserInfo(

name=form.name.data,

weight=form.weight.data,

height=form.height.data,

age=form.age.data,

gender=form.gender.data,

physical\_activity=form.physical\_activity.data,

user\_id=1 # Aqui você deve obter o ID do usuário logado

)

db.session.add(user\_info)

db.session.commit()

tdee = calc\_tdee(form.name.data, form.weight.data, form.height.data, form.age.data, form.gender.data, form.physical\_activity.data)

meals = generate\_meals(tdee)

return render\_template('results.html', meals=meals)

return render\_template('userinfo.html', title='Informações de Usuário', form=form)

Resumo.

Configuração do Banco de Dados: A configuração do banco de dados SQLite é feita em app.py com a variável SQLALCHEMY\_DATABASE\_URI.

Modelos de Banco de Dados: Os modelos User e UserInfo definem as tabelas user e user\_info no banco de dados.

Informações do Usuário: Quando um usuário envia o formulário de informações pessoais, esses dados são salvos na tabela user\_info.

As informações do usuário estão sendo registradas e armazenadas no banco de dados SQLite configurado no aplicativo Flask.

**8 TELAS.**

Mostraremos as imagens do nosso software abaixo:

# FIGURA 17 – TELA INICIAL.

A screenshot of a computer

Description automatically generated

FONTE: O PRÓPRIO AUTOR.

# FIGURA 18 – TELA DE LOGIN.

A screenshot of a computer

Description automatically generated

FONTE: O PRÓPRIO AUTOR.

# FIGURA 19 – TELA DE ERRO.

A screenshot of a computer

Description automatically generated

FONTE: O PRÓPRIO AUTOR.

# FIGURA 20 – TELA DE ERRO.

A screenshot of a computer

Description automatically generated

FONTE: O PRÓPRIO AUTOR.

# FIGURA 21 – TELA DE DADOS.

A screenshot of a computer

Description automatically generated

FONTE: O PRÓPRIO AUTOR.

# FIGURA 22 – TELA DE EXIBIÇÃO DE ALIMENTOS/HORÁRIOS.

A close-up of a paper

Description automatically generated

FONTE: O PRÓPRIO AUTOR.

**9 CONCLUSÃO.**

Ao longo deste projeto, desenvolvemos um sistema abrangente de recomendação de planos alimentares personalizados, utilizando uma variedade de tecnologias e metodologias para garantir sua eficácia e funcionalidade. A partir da análise das necessidades dos usuários e das especificidades do contexto, delineamos uma solução que visa fornecer orientações nutricionais precisas e adaptadas às características individuais de cada pessoa.

Utilizando uma abordagem híbrida, combinamos SQLite e listas em Python para armazenar e manipular os dados de forma eficiente. O SQLite foi empregado para gerenciar com segurança as informações dos usuários, incluindo nome, peso, altura, idade e nível de atividade física. Esses dados são essenciais para calcular as necessidades calóricas e nutricionais de cada indivíduo, permitindo assim a personalização das recomendações alimentares.

Por sua vez, as listas em Python foram utilizadas para gerar e manipular as recomendações de alimentação de forma flexível e dinâmica. Com base nas informações fornecidas pelo usuário, como suas preferências alimentares e restrições dietéticas, o sistema gera planos alimentares detalhados para diferentes refeições ao longo do dia, incluindo café da manhã, lanches, almoço, jantar e lanches noturnos.

Além disso, adotamos uma metodologia ágil de desenvolvimento, permitindo uma abordagem iterativa e colaborativa ao longo do processo. Isso nos possibilitou adaptar o sistema às necessidades emergentes dos usuários e realizar melhorias contínuas ao longo do tempo.

O resultado final é um software robusto e eficiente, capaz de fornecer recomendações alimentares personalizadas que visam promover uma alimentação saudável e equilibrada. Com essa solução, buscamos contribuir para o bem-estar e a qualidade de vida dos usuários, fornecendo-lhes ferramentas úteis e acessíveis para cuidar da sua saúde através da alimentação.

**10 MANUAL DO USUÁRIO.**

Bem-vindo ao Sistema de Recomendação de Planos Alimentares Personalizados. Este manual foi elaborado para ajudá-lo a utilizar todas as funcionalidades disponíveis no sistema de forma eficiente e satisfatória.

10.1 CADASTRO DE USUÁRIO.

* Para começar a usar o sistema, você precisa se cadastrar. Clique no botão "Cadastro" na página inicial.
* Preencha os campos solicitados, incluindo seu nome de usuário e senha desejada.

FIGURA 23 – MANUAL USUARIO.

A screenshot of a computer

Description automatically generated

FONTE: O PRÓPRIO AUTOR.

10.2 LOGIN.

* Após o cadastro, você poderá fazer o login no sistema utilizando o nome de usuário e senha cadastrados.
* Clique no botão "Login" na página inicial e insira suas credenciais.

# FIGURA 24 – TELA DE LOGIN.

A screenshot of a computer

Description automatically generated

FONTE: O PRÓPRIO AUTOR.

10.3 DASHBOARD.

* Após fazer o login, você será direcionado para o dashboard, onde encontrará várias opções disponíveis.
* No dashboard, você poderá visualizar suas informações que deve preencher de acordo com suas características.

FIGURA 25 – DASHBOARD.

A screenshot of a computer

Description automatically generated

FONTE: O PRÓPRIO AUTOR.

10.4 RECOMENDAÇÕES DE ALIMENTAÇÃO.

* No menu do dashboard, você encontrará a seção de "Recomendações de Alimentação".
* Nesta seção, você poderá visualizar os planos alimentares personalizados para cada refeição do dia, incluindo café da manhã, lanches, almoço, jantar e lanches noturnos.

# FIGURA 26 – REFEIÇOÊS DIARIAS.

A close-up of a paper

Description automatically generated

FONTE: O PRÓPRIO AUTO.

REFERÊNCIAS

**AWARI**. Lista Python: aprenda a criar e manipular listas de forma única. Disponível em: <https://awari.com.br/lista-python-aprenda-a-criar-e-manipular-listas-de-forma-unica/>. Acesso em: 18 maio 2024.

**ESSENTIAL NUTRITION**. Taxa Metabólica Basal. Disponível em: <https://www.essentialnutrition.com.br/conteudos/taxa-metabolica-basal/>. Acesso em: 23 maio 2024.

**INOVALIZE**. Conheça 5 aplicativos para obter inspiração para seu design. Disponível em: <https://inovalize.com.br/conheca-5-aplicativos-para-obter-inspiracao-para-seu-design/>. Acesso em: 01 maio 2024.

MIRO. **O que é UML?**. Disponível em: <https://miro.com/pt/diagrama/o-que-e-uml/>. Acesso em: 20 maio 2024.

**TUA SAÚDE**. Calculadora Taxa Metabólica Basal. Disponível em: <https://www.tuasaude.com/calculadora/taxa-metabolica-basal/#google_vignette>. Acesso em: 03 maio 2024.

**LUCIDCHART**. O que é UML. Disponível em: <https://www.lucidchart.com/pages/pt/o-que-e-uml>. Acesso em: 05 maio 2024.