UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

DIEGO SOUZA DE MELLO AFFONSO

Uma comparação entre plataforma tradicional de desenvolvimento e no-code para a criação de uma aplicação multidisciplinar de monitoramento fitness no formato MVP

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação

Orientador: Profa. Dra. Renata Galante

Porto Alegre
2024

“Eba! Um sonho que virou realidade!”

— Lillia

RESUMO

Existem aplicativos de nutrição para nutricionistas e pacientes, e aplicativos com foco em musculação, mas nenhum que una esse dois pontos. Um app que permita profissionais nutricionistas e educadores físicos a se conectarem com pacientes tende a beneficiar e maximizar os resultados dos usuários - seja com o objetivo de emagrecer, hipertrofiar, ou qualquer outro que seja planejado junto a um profissional qualificado - visto que a transformação do corpo depende de diversas variáveis (dentre elas, alimentação e exercício físico). Esse trabalho busca preencher esse espaço vago trazendo duas propostas de solução mobile que permitam profissionais terem dados mais completos sobre seus pacientes e usuários unificarem a experiência fitness pessoal, a primeira utilizando uma plataforma no-code e a segunda uma plataforma tradicional de desenvolvimento mobile. Para a criação do aplicativo em plataforma no-code foi utilizada a plataforma Thunkable, enquanto no caso da plataforma tradicional foi utilizado o *framework* Flutter, e, após a construção das duas soluções, foi realizada uma comparação entre os resultados de cada uma, com uma pequena margem de vantagem para a aplicação criada em plataforma tradicional.

Palavras-chave: Monitoramento. academia. musculação. nutrição. aplicativo. mobile. plataforma no-code. MVP.

A comparison between traditional development platform and no-code to create an multidisciplinary fitness monitoring application in MVP format

ABSTRACT

There are nutrition apps for nutritionists and patients, and apps focused on bodybuilding, but none that combine these two points. An app that allows professional nutritionists and physical educators to connect with patients tends to benefit and maximize users' results - whether with the objective of losing weight, hypertrophy, or any other objective that is planned with a qualified professional - since the transformation of the body depends on several variables (among them, diet and physical exercise). This work seeks to fill this vacant space by bringing two mobile solution proposals that allows professionals to have more complete data about their patients and users to unify their personal fitness experience, the first using a no-code platform and the second using a traditional mobile development platform. For the creation of the application on a no-code platform, the Thunkable platform was used, while in the case of the traditional platform, the Flutter framework was used, and, after building the two solutions, a comparison was made between the results of each one, with a small margin of advantage for the application created on a traditional platform.

Keywords: Monitoring, gym, bodybuilding, nutrition, app, mobile, no-code platform, MVP.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ASQ	After Scenario Questionnaire
CLI	Command Line Interface
MVP	Minimum Viable Product
SUS	System Usability Scale
UFRGS	Universidade Federal do Rio Grande do Sul

SUMÁRIO

1 INTRODUÇÃO

Existem diversos aplicativos relacionados à musculação disponíveis no mercado, buscando disponibilizar treinos e gerar estatísticas sobre o progresso do usuários - assim como existem também (ainda que em menor número) aplicativos relacionados à nutrição e planos alimentares, com o objetivo de disponibilizar dietas e contar quantidade de macronutrientes e calorias consumidos. O problema com essas aplicações é que, em sua grande maioria, elas têm o intuito de substituir o profissional de educação física/nutrição, decisão essa que é desencorajada por especialistas, tendo em vista que estudos comprovam tanto que o treino com um educador físico dá mais resultado (McClaran, 2003), quanto que dietas sem acompanhamento nutricional podem ser perigosas (Moura et al., 2022). Ademais, são sempre aplicações distintas que o usuário precisa baixar e replicar seus dados e informações, criando dois quadros incompletos sobre seu condicionamento físico, visto que alimentação e exercícios são fatores complementares para uma boa forma física.

Apresentado o cenário, este trabalho tem como objetivo criar o MVP de uma aplicação que sirva para unir a experiência dos usuários e profissionais. O aplicativo busca a interação entre usuários profissionais e usuários comuns, apresentando funcionalidades que já existem no mercado - mas não em apenas um *software*. Assim sendo, é importante mostrar que há um espaço aberto no mercado para uma aplicação integrada entre treinos e planos alimentares para melhor monitorar a evolução do usuário e, para isso, os aplicativos já existentes hoje no mercado serão avaliados e comparados com a solução proposta.

Com a construção de um MVP em evidência, a oportunidade de comparar a criação desta aplicação através de uma plataforma no-code, modelo que cresce rapidamente no mercado, com uma forma mais tradicional de desenvolvimento mobile é possibilitada. Os MVPs desenvolvidos ao longo deste trabalho têm como proposta criar uma interface amigável tanto para o usuário comum, quanto para o usuário profissional de nutrição ou de educação física, que possibilite a execução de funcionalidades básicas desse tipo de aplicativo, requisitos que serão discutidos ao longo da monografia.

O trabalho está dividido em 8 capítulos, divididos da seguinte forma: o capítulo 2 apresenta as tecnologias utilizadas. O capítulo 3 apresenta os trabalhos relacionados. O capítulo 4 apresenta o desenvolvimento. O capítulo 5 apresenta a demonstração das aplicações. O capítulo 6 apresenta os experimentos e avaliações dos resultados. O capítulo 7 apresenta a conclusão. Por fim, o capítulo 8 apresenta os apêndices A e B que contêm

a avaliação heurística feita para teste com desenvolvedores e questionário utilizado para teste com usuários.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem como objetivo apresentar os conceitos e tecnologias empregados durante a construção do projeto.

2.1 Ferramentas de Desenvolvimento

Aqui serão apresentadas as ferramentas de desenvolvimento utilizadas no projeto.

2.1.1 Thunkable

Thunkable[1] é uma plataforma para desenvolvimento mobile que é divulgada como "criação de app sem código". Analisando objetivamente, entretanto, observa-se que o usuário da plataforma estará sim utilizando código - mas de uma maneira mais intuitiva, com blocos de código similares aos da famigerada linguagem de programação Scratch. Lançada em 2016, a aplicação possui duas interfaces principais: uma para a criação do design (front-end), com telas, botões, imagens, texto, entre outros, e uma para a criação dos blocos de código (back-end), com blocos de controle, blocos lógicos, blocos referentes ao dispositivo, entre outros. A plataforma ainda oferece download e publicação dos apps criados para Android, IOS e aplicações web.

2.1.2 Dart

Dart[2] é uma linguagem de programação desenvolvida pela Google que foi projetada para criar aplicativos de alto desempenho em diversas plataformas distintas. Apresentando uma sintaxe orientada a objetos, Dart oferece recursos como *garbage collection*, facilitando o desenvolvimento e a manutenção de código limpo, permitindo ao desenvolvedor focar mais em gerar o código do que em geri-lo - sem precisar se preocupar em administrar a memória. Sua principal característica é a compilação *just-in-time* e *ahead-of-time*, permitindo uma rápida execução em tempo de desenvolvimento. Amplamente utilizado para criar aplicativos móveis e web, o Dart é uma escolha popular entre desen-

[1] https://docs.thunkable.com
[2] https://dart.dev/guides

volvedores em busca de produtividade e desempenho, e costuma ser utilizado em conjunto com o kit de desenvolvimento de interface de usuário Flutter.

2.1.3 Flutter

Flutter[3] é um kit de desenvolvimento (*framework*) desenvolvido pela Google para a criação de interfaces de usuário. Projetado para facilitar a construção de aplicativos, a ferramenta permite a criação de aplicações nativas para Android, IOS, aplicações web e até mesmo aplicações de desktop através de um código-fonte único. Ao utilizar a linguagem Dart no *framework*, o programador pode aproveitar os recursos dos widgets personalizáveis em conjunto com a abordagem de "*hot reload*" - que permite atualizar a aplicação sem perder o estado em que se encontra - para acelerar o ciclo de desenvolvimento. Sendo um software de código aberto e com desempenho eficiente, é escolhido cada vez mais por desenvolvedores, com pesquisas recentes apontando mais de 2 milhões de usuários do kit[4].

2.2 Ferramentas auxiliares

Aqui serão apresentadas as ferramentas auxiliares utilizadas no projeto.

2.2.1 Visual Studio Code

Popularmente conhecido como "VS Code"[5], é um editor de código-fonte desenvolvido pela Microsoft e amplamente utilizado pela comunidade de programadores mundialmente. Possui suporte para diversas linguagens de programação e *frameworks*, possibilitando o download de extensões para complementar a experiência. Com uma interface intuitiva e inúmeros recursos de edição, o editor auxilia na produtividade e eficiência de desenvolvedores. No caso do desenvolvimento em Flutter, as extensões "Dart" e "Flutter" são necessárias.

[3] https://docs.flutter.dev

[4] https://www.linkedin.com/pulse/flutter-usage-statistics-2023-unveiling-rise-joe-shestak/ (acesso em fevereiro de 2024)

[5] https://code.visualstudio.com/docs

2.2.2 Github

GitHub[6] é uma plataforma de desenvolvimento colaborativo de software baseada em nuvem, conhecida por oferecer um sistema de controle de versão distribuído usando o Git, o que permite que desenvolvedores trabalhem de maneira mais eficiente. Fundada em 2008 por Chris Wanstrath, PJ Hyett e Tom Preston-Werner, o GitHub oferece recursos como rastreamento de problemas, solicitações de *pull* e integração contínua, facilitando o gerenciamento de projetos. Sua interface intuitiva promove a transparência e a eficiência no desenvolvimento de software, tornando-o uma escolha popular entre desenvolvedores de todas as habilidades e experiências. Além disso, o Git, ferramenta de versionamento de código, simplifica significativamente o desenvolvimento, permitindo aos usuários criar, usar e modificar ramificações (*branches*) rapidamente, o que possibilita experimentações e refinamento de mudanças locais antes da incorporação ao código principal.

2.3 Conceito teórico

Aqui serão apresentados os conceito teóricos utilizados no projeto.

2.3.1 *Minimum Viable Product* (MVP)

O MVP, ou Mínimo Produto Viável[7], representa o produto/serviço com apenas *features* suficientes para satisfazer clientes iniciais, focando em realizar rapidamente a criação e lançamento - buscando *feedback* para futuro desenvolvimento do produto. É uma prática comum no empreendedorismo que busca validar velozmente a viabilidade de uma ideia, priorizando recursos essenciais e minimizando custos. Com um MVP, empresas podem testar hipóteses de mercado de forma eficiente, direcionando seus recursos para áreas que agregam mais valor aos usuários - os requisitos - e resultando em um produto final mais sólido e adaptado às necessidades do mercado.

[6] https://docs.github.com/pt
[7] RIES, E. **The Lean Startup**. New York: Crown Business, 2011.

3 TRABALHOS RELACIONADOS

Este capítulo tem como objetivo apresentar os trabalhos relacionados com este projeto e realizar uma análise comparativa entre esses e a solução proposta.

3.1 Aplicações semelhantes

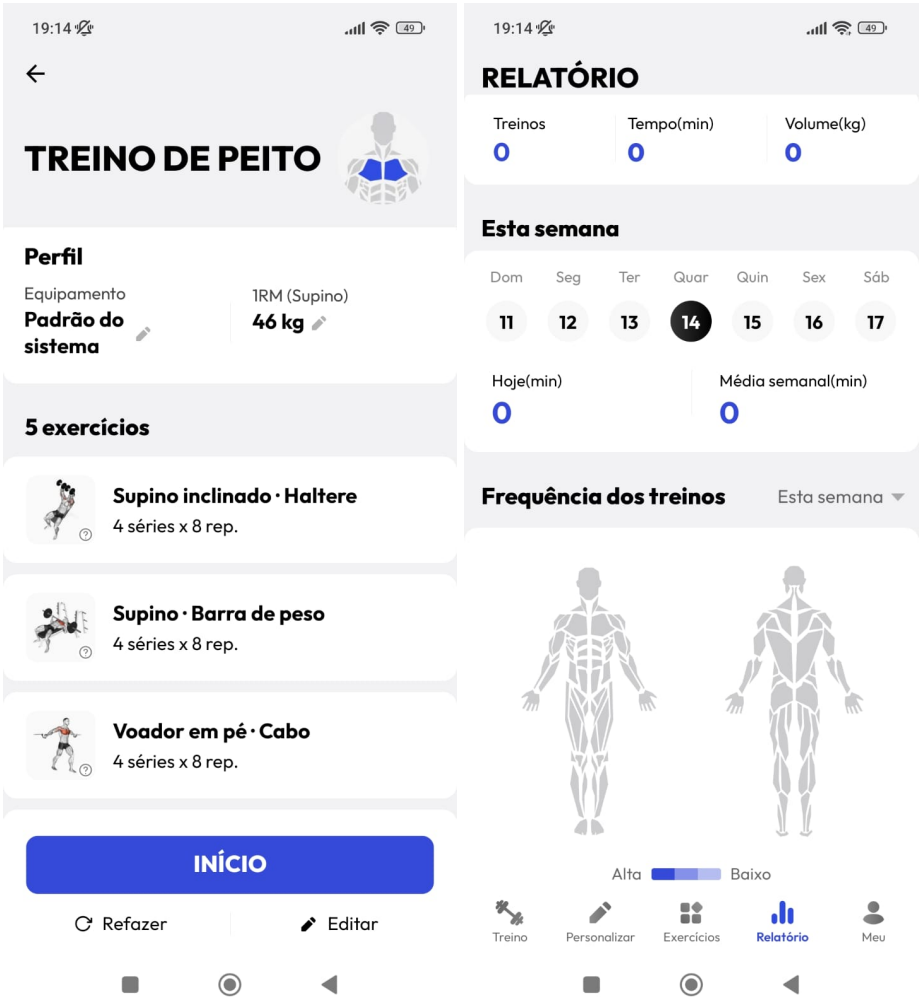
Foram selecionados quatro aplicativos que possuem semelhanças com a solução proposta neste trabalho. Como o diferencial da aplicação aqui apresentada é a multi-disciplinaridade, foi escolhido apresentar dois aplicativos com a temática de treino de academia, e dois aplicativos com a temática de nutrição. Como será visto, muitas das funcionalidades descritas em cada aplicação semelhante são similares as deste trabalho, motivo pelo qual elas foram escolhidas dentre tantas nas lojas de aplicativos.

3.1.1 Treino de Academia

Treino de Academia[1], ilustrado na Figura 3.1, é um aplicativo para Android e IOS que busca orientar o usuário com rotinas clássicas personalizadas criadas por especialistas para atender diferentes objetivos. Além disso, possui personalização de treinos, registro de execução de treino e gráficos estatísticos apresentando os dados monitorados. É um dos aplicativos mais completos para área de musculação, ostentando alta nota tanto na Play Store quanto na App Store, focado totalmente para o uso individual do usuário.

[1] https://play.google.com/store/apps/details?id=gymworkout.gym.gymlog.gymtrainer&hl=pt_BR (acesso em fevereiro de 2024)

Figura 3.1: Telas mostrando funcionalidades do app



(a) Tela mostrando treino (b) Tela mostrando relatório

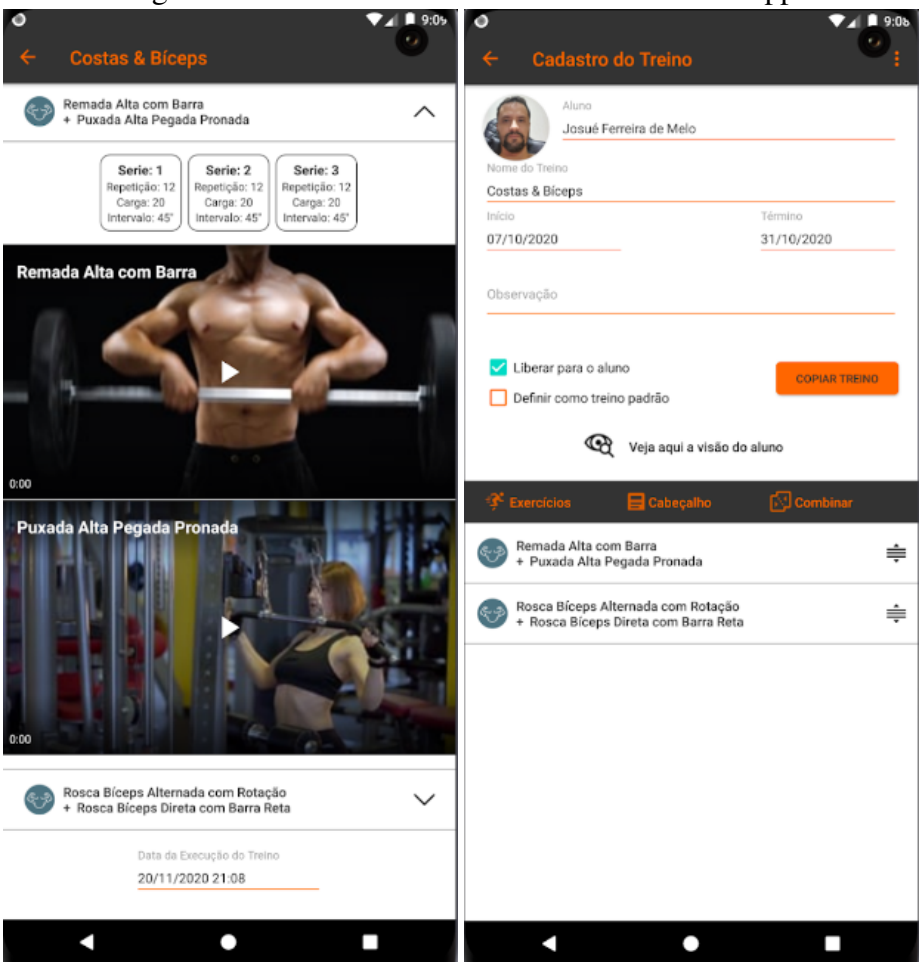
Fonte: Gerado pelo autor

3.1.2 Avaliação Física PRO

Avaliação Física PRO[2], ilustrado na Figura 3.2, também é um aplicativo para Android e IOS ligado à área de exercícios físicos, mas dessa vez focado na interação entre instrutor/treinador e aluno. Apresenta rotina de exercícios para alunos com possibilidade de *feedback* para cada treino, além da função de resultados estatísticos das avaliações de alunos. Possui a possibilidade de entrar como aluno ou como instrutor no aplicativo.

[2]https://play.google.com/store/apps/details?id=com.codeapp.avaliacaofisica&hl=pt_BR (acesso em fevereiro de 2024)

Figura 3.2: Telas mostrando funcionalidades do app



(a) Tela do paciente mostrando treino (b) Tela do instrutor mostrando treino

Fonte: Gerado pelo autor

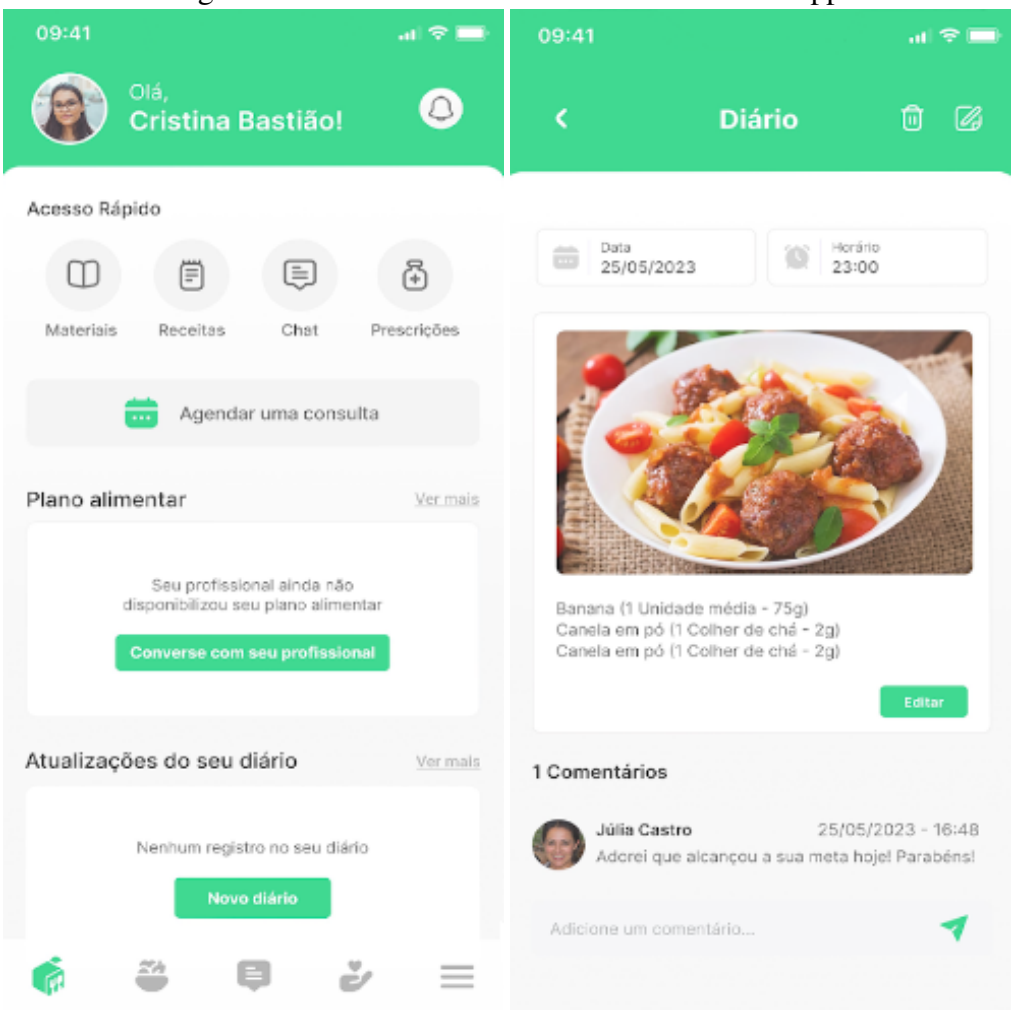
3.1.3 Dietbox

Dietbox[3], ilustrado na Figura 3.3, e Dietbox para profissionais[4], ilustrado na Figura 3.4, são aplicativos complementares utilizados respectivamente por usuários e nutricionistas. O app do usuário possui funcionalidades como diário de alimentação, plano alimentar, evolução de antropometria e chat. O app do profissional possui acesso ao plano alimentar de clientes, criação de agendamentos de consultas, acompanhamento dos diários alimentares e chat.

[3]https://play.google.com/store/apps/details?id=com.craftbox.dietbox&hl=pt_BR (acesso em fevereiro de 2024)

[4]https://play.google.com/store/apps/details?id=com.craftbox.dietboxprofissional&hl=pt_BR (acesso em fevereiro de 2024)

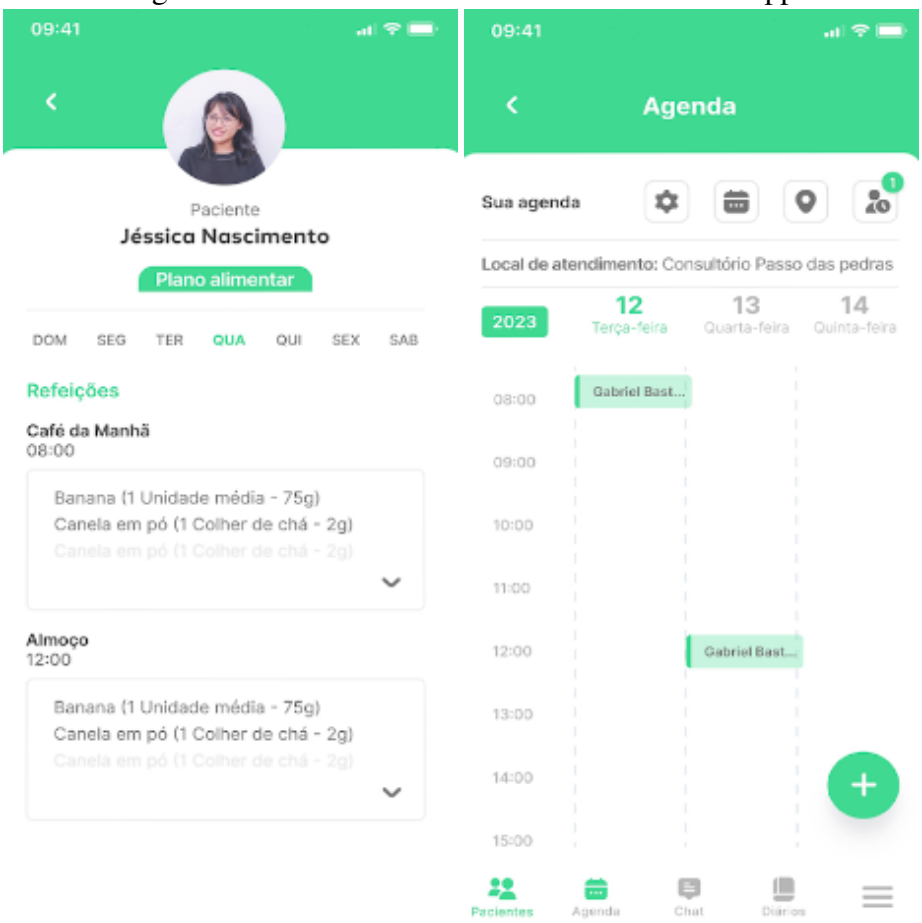
Figura 3.3: Telas mostrando funcionalidades do app



(a) Tela mostrando menu do paciente (b) Tela mostrando diário alimentar

Fonte: Gerado pelo autor

Figura 3.4: Telas mostrando funcionalidades do app



(a) Tela mostrando paciente (b) Tela mostrando agendamentos

Fonte: Gerado pelo autor

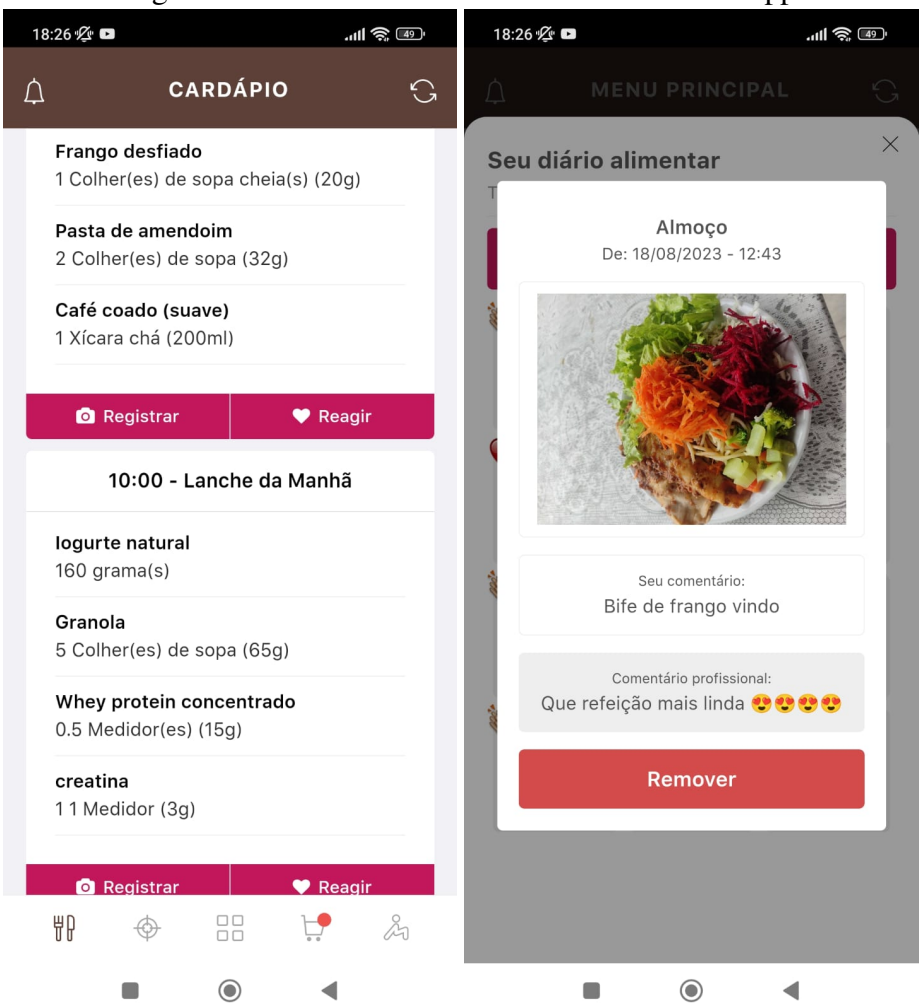
3.1.4 WebDiet

WebDiet[5], ilustrado na Figura 3.5, e WebDiet Pro[6], ilustrado na Figura 3.6, são também aplicativos complementares utilizados respectivamente por usuários e nutricionistas, sendo considerados como concorrentes do software Dietbox. O app do usuário também possui funcionalidades como diário de alimentação, plano alimentar, evolução de antropometria e chat, além de adicionar a confirmação de metas e evolução fotográfica (fotos do corpo). O app do profissional possui acesso às mesmas funcionalidades que o app Dietbox para profissionais.

[5]https://play.google.com/store/apps/details?id=br.com.webdiet.webdiet&hl=pt_BR (acesso em fevereiro de 2024)

[6]https://play.google.com/store/apps/details?id=br.com.webdiet.webdietpro&hl=pt_BR (acesso em fevereiro de 2024)

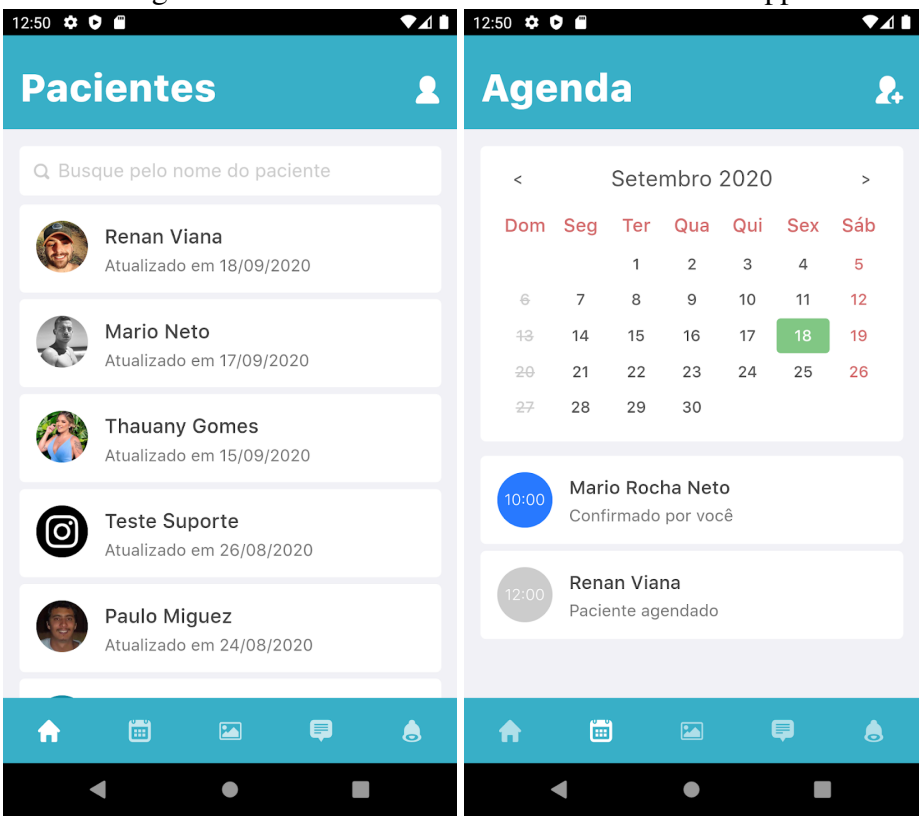
Figura 3.5: Telas mostrando funcionalidades do app



(a) Tela mostrando plano alimentar (b) Tela mostrando diário alimentar

Fonte: Gerado pelo autor

Figura 3.6: Telas mostrando funcionalidades do app



(a) Tela mostrando pacientes (b) Tela mostrando agendamentos

Fonte: Gerado pelo autor

3.2 Análise comparativa

O enfoque deste trabalho não é desenvolver uma nova funcionalidade que não esteja presente nos aplicativos apontados, visto que as ferramentas hoje utilizadas já cumprem satisfatóriamente as necessidades de usuários e profissionais, mas sim unir alguns desses recursos em apenas uma aplicação.

A seguir, na Tabela 3.1, é apresentada uma comparação entre as aplicações apresentadas e a aplicação aqui proposta (intitulada “TCC” na tabela), comparando suas funcionalidades.

Tabela 3.1: Comparação de funcionalidades

	Treino de academia	*Avaliação Física PRO*	*WebDiet*	*Dietbox*	*TCC*
Treino para usuário	Sim	Sim	Não	Não	Sim
Cardápio para usuário	Não	Não	Sim	Sim	Sim
Interação com nutricionista	Não	Não	Sim	Sim	Sim
Interação com educador físico	Não	Sim	Não	Não	Sim
Estatísticas de evolução	Sim	Sim	Sim	Sim	Não
Feedback do usuário	Não	Sim	Sim	Sim	Sim
Chat	Não	Sim	Sim	Sim	Não

Fonte: O Autor

Como esperado, não existe uma aplicação universal para nutricionistas e educadores físicos, fazendo com que um usuário precise de no mínimo dois aplicativos complementares para conseguir dados mais completos. Além disso, o profissional não tem um quadro completo da rotina/comportamento do paciente, visto que faltam informações presentes somente no aplicativo do outro profissional que está acompanhando o usuário.

Apesar de todos, exceto a presente aplicação, possuírem estatísticas de evolução, essas são distintas entre os aplicativos de academia e os de nutrição - sendo que normalmente os de nutrição contêm dados antropométricos, enquanto os de academia contêm dados relativos aos pesos que o usuário está utilizando e número de repetições. Ademais, a funcionalidade de chat não foi contemplada na solução criada por existirem aplicativos de chat já consolidados, como WhatsApp, que tornariam essa *feature* desnecessária em um MVP.

4 METODOLOGIA

Este capítulo tem como objetivo apresentar a metodologia, passando pela importância de criação de um MVP, escolha dos representantes da plataforma no-code e tradicional, requisitos para a aplicação, características dos representantes e concluindo com uma comparação entre os dois métodos.

4.1 Visão Geral

Como para o propósito desse trabalho foi necessário escolher uma plataforma no-code e uma plataforma tradicional de desenvolvimento para a comparação entre as duas soluções finais, essas escolhas irão representar o conjunto em que elas estão englobadas. Assim sendo, nesse trabalho será usado de forma alternada “plataforma no-code” e “Thunkable”, assim como será usado de forma alternada “plataforma tradicional” e “Flutter”.

4.1.1 Importância de um MVP

O modelo MVP já é padrão na indústria de desenvolvimento de software, fazendo com que desenvolvedores possam evitar trabalho longo - e possivelmente desnecessário - cortando custos para empresas que podem melhor realocar posteriormente seu capital, como em *features* novas que clientes iniciais peçam, mas que não estavam no planejamento original. A validação da viabilidade do projeto em um mercado real pode rapidamente definir se o projeto irá ser continuado ou se não há demanda para ele, através do *feedback* dos usuários que utilizarem a primeira versão, e também pode gerar *insights* sobre pontos fortes do produto.

Para o caso da proposta deste trabalho, o MVP poderia validar que usuários teriam interesse em um aplicativo com uma abordagem multidisciplinar entre academia e alimentação antes que todas as funcionalidades fossem implementadas. Para isso, foi escolhido desenvolver um MVP na modalidade protótipo, onde é realizada a criação de um protótipo funcional para o produto/serviço para efetuar testes necessários com os usuários, com o intuito de fazer com que naveguem e explorem as interfaces da aplicação. Esse modelo de MVP, apesar de ser um dos mais custosos por o protótipo se assemelhar

com o produto final, é o que oferece maior quantidade de retorno dos clientes.

4.1.2 Escolha da plataforma no-code

Para a escolha da plataforma no-code era necessário que o foco do plataforma fosse a criação de aplicações *mobile*, tendo em vista a proposta deste trabalho. O Thunkable não só preenche esse pré-requisito, como já foi utilizado previamente pelo autor desse texto, facilitando a escolha da plataforma. Ademais, a seguir, são apresentados alguns motivos para ratificar a escolha do Thunkable:

- alto número de usuários divulgados pelo próprio Thunkable: 10 milhões de apps criados, 3.5 milhões de criadores;
- extensa documentação;
- comunidade ativa no fórum, havendo mais de 14 mil tópicos somente de questões relacionadas ao uso da aplicação; e
- canal no YouTube com mais de uma centena de vídeos tutoriais.

4.1.3 Escolha da plataforma tradicional

Para a escolha da plataformar tradicional, decidiu-se por selecionar uma plataforma que pudesse gerar aplicativo tanto para Android, quanto para IOS, automaticamente desclassificando linguagens como Kotlin e Swift. Flutter e React Native costumam ser os primeiros nomes lembrados quando se tem essa necessidade, por precisar de apenas uma base de código-fonte, mas o Flutter foi escolhido porque:

- possui maior desempenho (enquanto React Native utiliza JavaScript, Flutter é compilado utilizando a biblioteca arm C/C++);
- possui documentação mais estruturada, criada pela Google; e
- renderiza todos os componentes em sua própria tela e, devido a isso, atualizações de componentes não têm nenhum impacto em aplicativos Flutter, mas têm em aplicativos React Native.

Além disso, Flutter oferece uma ampla variedade de widgets personalizáveis, permitindo que os desenvolvedores criem interfaces visualmente atraentes que se assemelham de perto a aplicativos nativos, possuindo também um recurso de *hot reload* que reduz significativamente o tempo de desenvolvimento, fornecendo feedback instantâneo sobre as alterações no código.

4.2 Requisitos

Tendo em vista que os aplicativos já existentes para os profissionais (como Dietbox para profissionais e WebDiet Pro) delegam funcionalidades como cadastro de pacientes e criação/edição de planos alimentares para uma aplicação web, foi decidido que a parte da gerência dos treinos e dietas de pacientes poderia ser melhor manipulada pelos profissionais através de um método que a maioria já utiliza - planilhas eletrônicas. Para isso, foi pensado em utilizar a plataforma online Google Sheets para registrar essas informações de maneira que facilitasse o trabalho dos especialistas, incorporando mudanças feitas na tabela junto à aplicação. Essa ideia, no entanto, não foi validada junto a profissionais e a escolha de como essa parte irá integrar o sistema ficará para trabalho futuro, ficando assim fora do escopo do MVP.

Assim, inicialmente - com base na experiência própria do autor e em conversas breves com um profissional de nutrição e um profissional de educação física - foram definidos requisitos necessários para usuários e profissionais, dispostos da seguinte maneira:

1. Usuário Comum
    - Visualizar plano alimentar
    - Visualizar treinos
    - Visualizar e registrar metas
2. Usuário Profissional
    - Visualizar e agendar consultas
    - Visualizar pacientes

Ademais, foram também definidos requisitos desejáveis, que seriam:

1. Usuário Comum

- Visualizar diário alimentar
- Visualizar histórico de treinos
- Registrar avaliação pessoal e visualizar seu histórico
- Chat
- Editar perfil

2. Usuário Profissional

- Cadastrar pacientes
- Chat
- Editar perfil

Após a criação do MVP em plataforma no-code, verificou-se também a necessidade do profissional visualizar o plano alimentar e treino dos pacientes. Além disso, as funcionalidades de chat, editar perfil de usuário e editar perfil de profissional acabaram virando requisitos adicionais, desnecessários no sentido de um MVP, visto que o cadastro de pacientes e seus respectivos dados não estão contemplados no trabalho, e que o aplicativo WhatsApp poderia ser utilizado como chat enquanto outras funcionalidades mais importantes são desenvolvidas.

Após a criação do MVP em plataforma tradicional, constatou-se que o requisito de visualização de agendamentos por parte do usuário comum também seria necessário, mas acabou não entrando no escopo desse trabalho por sua identificação tardia.

Após a definição primária dos requisitos, foi decidido que a primeira implementação do MVP seria em Thunkable. Esse fato se deu tanto por plataformas no-code em geral apresentarem menor tempo de desenvolvimento[1], quanto por o autor já ter conhecimento de Thunkable e não de Flutter, diminuindo a barreira de entrada.

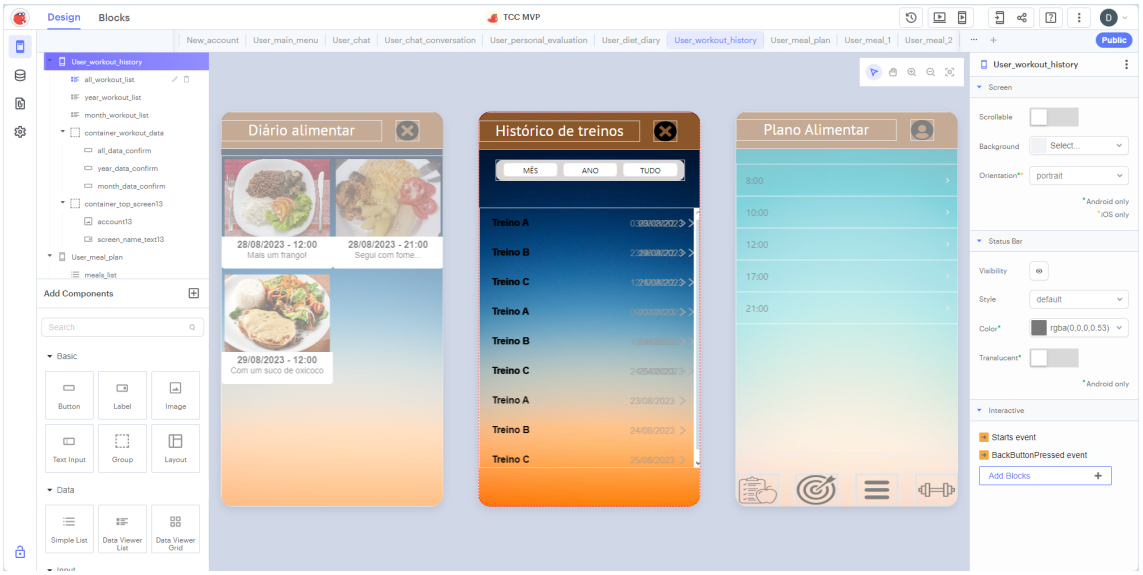
4.3 Thunkable

O Thunkable apresenta duas interfaces principais: Design e Blocks, que cuidam respectivamente do *front-end* e do *back-end* do aplicativo. A parte de desenvolvimento em Thunkable é *fullstack*, alternando entre as duas interfaces para a criação das telas,

[1] https://www.weforum.org/agenda/2023/08/no-code-platforms-speed-digitalization/ (acesso em fevereiro de 2024)

primeiro adicionando componentes no design e depois programando seu comportamento nos blocos. Podemos utilizar um modo chamado *Web Preview* para visualizar a aparência e comportamento das telas, simulando o uso de um dispositivo e servindo para teste. Na sequência, o comportamento de cada interface é elaborado.

Figura 4.1: Interface de design da plafatorma Thunkable



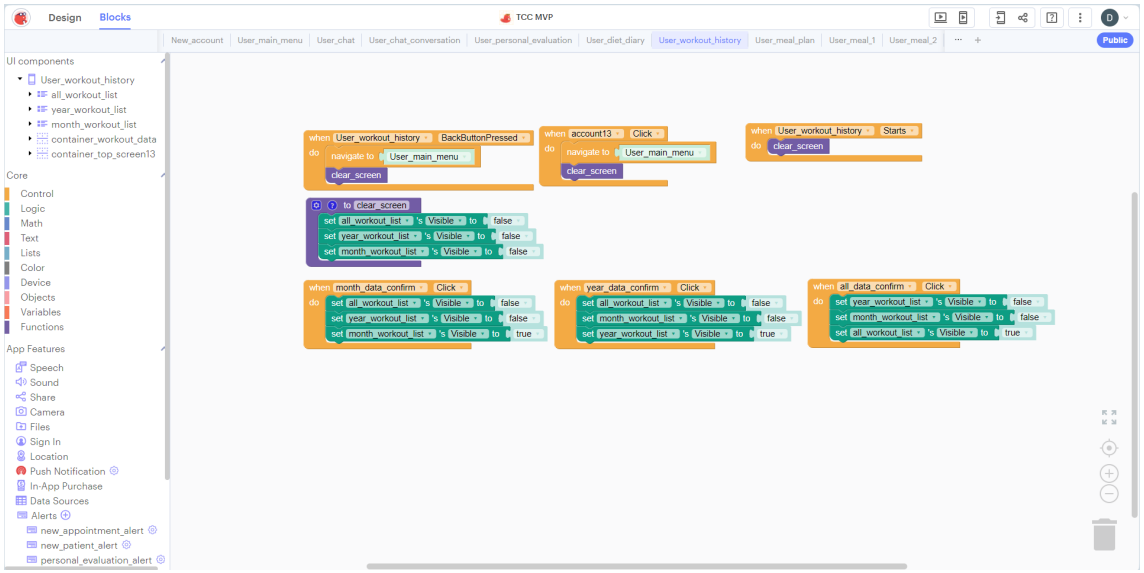
Fonte: Gerado pelo autor

Começando pelo design, cuja Figura 4.2 ilustra, podemos observar que a interface é composta por um conjunto central de telas, onde cada uma representa uma tela do aplicativo, uma coluna na esquerda dividida entre *Component Tree*, que representa os níveis e subníveis de cada componente presente em cada tela, e *Add Components* que dispõe dos diversos tipos de componentes a serem adicionados a tela, como textos, botões, listas, canvas, *web viewers*, entre outros. Ademais, ainda mais a esquerda há uma coluna estreita que permite acessar os dados - que podem ser criados em tabelas locais, ou em plataformas como Google Sheets e Airflow - os assets, como imagens, e configurações do app, para alterar seu nome, versionamento e ícone, entre outras coisas. Por último, na direita, há uma coluna que server para alterar o comportamento inicial de um componente selecionado (como modificar o texto inicial de um botão, ou definir a visibilidade de uma lista).

Focando nos componentes, vemos desde componentes básicos e costumeiros, como texto, entrada de texto e botões, até componentes mais complexos, como canvas, *web viewer* e alguns mais utilizados no projeto como *data viewer list* e *data viewer grid*, e esses elementos são posicionados na tela selecionada através do método *drag and drop*, que consiste em arrastar o componente da aba *Add Components* até o lugar desejado. Explo-

rando mais o componente *data viewer list*, que foi o mais utilizado no trabalho, observamos que seu caso de uso é para quando gostaríamos de uma lista com dados dinâmicos já registrados em um local, como uma planilha eletrônica. Com ele, podemos definir qual a fonte dos dados, qual a tabela a ser utilizada (caso tenha mais de uma na mesma planilha), o modelo a ser utilizado (somente título; título e subtítulo; título, subtítulo e imagem; etc) e de que coluna da tabela cada dado deve vir. Esse é somente o comportamento inicial, o resto será visto na parte dos blocos do componente a seguir.

Figura 4.2: Interface de blocos da plafatorma Thunkable



Fonte: Gerado pelo autor

Indo para os blocos, podemos observar sua interface na figura 4.2, que é composta por um grande quadro branco central, unido a uma coluna posicionada à esquerda contendo os blocos de código. Essa coluna é dividida em 3 partes: a primeira contendo blocos relacionados aos componentes presentes na tela, a segunda contendo os blocos mais básicos e essenciais a qualquer aplicação - como blocos lógicos e de controle - e a terceira contendo *features* especiais, como localização e uso da câmera.

Podemos utilizar funções de controle como *if*, *for* e *while*, além da função de navegação entre telas. Esses blocos são utilizados normalmente como blocos mais externos, empregando blocos lógicos como condição de parada e com blocos matemáticos, de texto e de variáveis dentro do primeiro bloco, fazendo alterações no comportamento do aplicativo. Utilizando novamente a Figura 4.2, que é a parte do *back-end* da tela presente na Figura 4.1, podemos observar um bloco de função "*Clear Screen*" criado pelo autor para tornar invisíveis as três listas da tela, utilizando blocos próprios de cada *data viewer list* (que podem ser acessados ao clicar em cada componente na primeira parte da coluna).

Essa função é chamada toda vez que alguma navegação para fora da tela é realizada, fazendo com que toda vez que a interface seja novamente acessada, nada apareça. Quando um dos botões é pressionado, entretanto, é definida qual lista irá aparecer.

Por fim, é de extrema importância ressaltar que o Thunkable possui planos com diferentes preços. Para este trabalho, foi utilizado o único plano gratuito, que só possibilita dois downloads do aplicativo por mês. Abaixo uma comparação entre os distintos planos, na Tabela 4.1.

Tabela 4.1: Comparação de planos da plataforma Thunkable

	Free plan	*Starter plan*	*Pro plan*	*Business plan*	*Team plan*
Preço	Gratuito	$15/mês	$45/mês	$200/mês	$500/mês
Limite de projetos	10	20	Sem limite	Sem limite	Sem limite
Projetos privados	0	1	50	Sem limite	Sem limite
Números de versões (*branches*)	1	1	10	50	50
Número de downloads	2/mês	25/mês	Sem limite	Sem limite	Sem limite
Colaboração	Não	Não	Não	Não	Sim
Web Apps	0	0	2	5	Sem limite

Fonte: O Autor

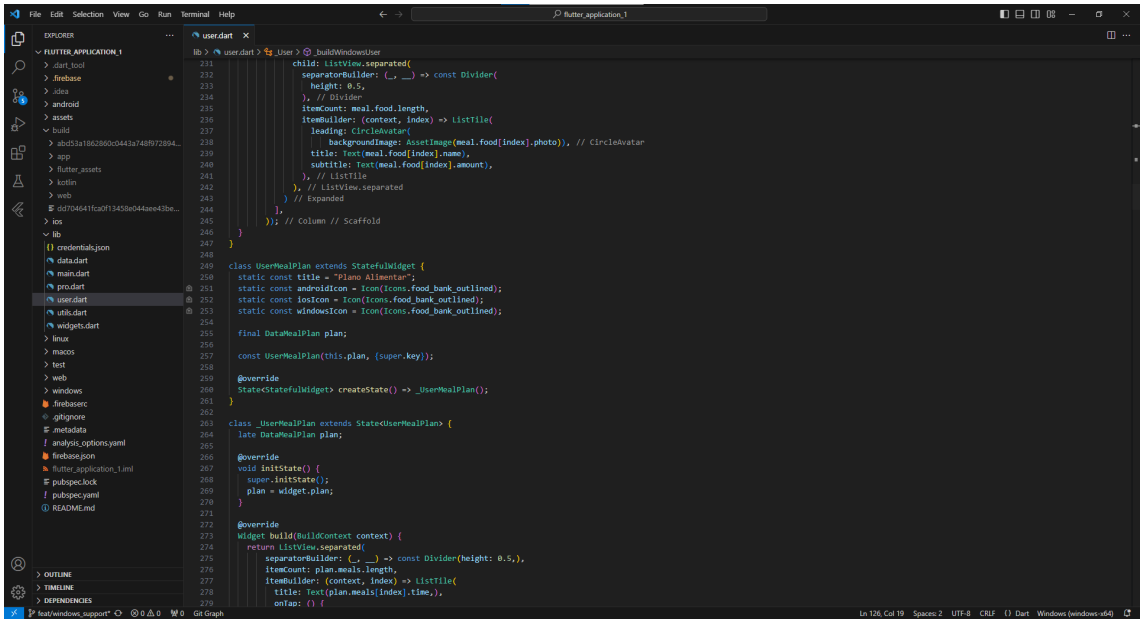
4.4 Flutter

A construção em Flutter se aproxima muito a construção em outras plataformas tradicionais de desenvolvimento mobile, no sentido de ter uma base textual de código, poder alterar o código em qualquer editor de texto que o programador preferir, e poder utilizar ferramentas distintas já empregadas entre desenvolvedores, como Git e CLI.

A linguagem Dart utiliza classes para quase tudo no desenvolvimento, tendo muitos componentes desejáveis em aplicativos já prontos para utilização, não só básicos como botões, mas também complexos como *Scaffold* com *AppBar* e *BottomNavigator* (as barras de navegação superior e inferior em aplicativos). O desenvolvimento em Flutter, ainda, se baseia muito no uso de *widgets* em diversos modelos, além de *widgets* customizáveis pelo programador. Além disso, ao utilizar distintos *WidgetBuilder*, é possível a construção de código para sistemas diferentes (como Android e IOS) ao mesmo tempo - podendo ainda

customizar diferentes ícones, imagens e formas de uso para cada aplicação gerada. Na Figura 4.3 podemos ver uma parte do código ilustrada.

Figura 4.3: Exemplo de código em Flutter



Fonte: Gerado pelo autor

4.5 Comparação

Existem algumas comparações a serem feitas, com vantagem para os dois lados, então não há uma obviedade de qual plataforma escolher para desenvolver. Dito isso, abaixo serão apresentadas algumas dessas comparações, e nos próximos dois capítulos essa discussão será continuada.

- **Tempo:** o tempo de desenvolvimento destoou muito entre os dois MVPs. Ainda que o tempo não tenha sido cronometrado para cada criação, o autor estima que gastou 20 horas na construção do MVP no-code e 50 horas no MVP tradicional - isso levando em consideração que o segundo se baseou muito no primeiro, onde certas partes deveriam ser apenas replicadas.

- **Barreira de entrada:** antes de discutir a barreira de entrada, é necessário frisar que o autor já havia trabalhado com Thunkable, mas não com Flutter. Dito isso, é inegável que a interface de desenvolvimento do Thunkable é muito mais amigável, especialmente por visar pessoas não programadoras como a maior parte do público-alvo, - a barreira de entrada do Dart é maior no caso de um programador e quase

intransponível no caso de uma pessoa comum.

- **Tamanho da aplicação:** é importante comparar também o tamanho do aplicativo final para Android - enquanto o app finalizado em Flutter ficou com 41,92MB, o app finalizado em Thunkable ficou com 154MB - mais de 3.5 vezes maior.

- **Limitações:** o Thunkable apresentou limitações que atrapalharam durante o desenvolvimento do MVP. Enquanto a maioria das dificuldades em Flutter eram causadas pela falta de conhecimento do autor e maior curva de aprendizado, em Thunkable eram causadas por limites da plataforma. Como exemplo temos a complexidade de criar linha e figuras geométricas na tela, sendo necessário o uso do componente Canvas para definir as coordenadas de cada pontos (coordenadas essas que não necessariamente vão se espelhar entre o modo *Web Preview* e o dispositivo real). Ademais, não é possível alterar qual base de dados será passada para cada *data viewer list* e *data viewer grid*, impossibilitando que um desses componentes receba como entrada uma variável (o que criou, por exemplo, a necessidade de se fazer uma tela para cada opção de horário do plano alimentar, como será visto no próximo capítulo). Por último, componentes que eram básicos em Flutter, como as barras de navegação superior e inferior, precisaram ser criados do início pelo autor.

A seguir, na Tabela 4.2, é apresentado um sumário contrastando as duas abordagens.

Tabela 4.2: Sumário das soluções

Plataforma	Tempo gasto estimado	Barreira de entrada	Tamanho da aplicação
No-code	20h	Baixa	154MB
Tradicional	50h	Média	42MB

Fonte: O Autor

5 DEMONSTRAÇÃO

Este capítulo tem como objetivo apresentar as funcionalidades dos MVPs construídos através de uma demonstração visual das telas junto com a explicação de seus fluxos. A demonstração se dará interpolando o MVP no-code e o MVP tradicional para o usuário comum e para o usuário profissional.

5.1 Visão Geral

A aplicação aqui apresentada busca preencher um espaço aberto no mercado de aplicativos para ser usado tanto por clientes de nutricionistas e de educadores físicos, como pelos próprios profissionais. O aplicativo busca unificar recursos multidisciplinares - tanto de educação física, quanto de educação alimentar - para gerar uma experiência mais completa aos usuários. Funcionalidades como visualização de plano alimentar e de treinos, no caso do usuário comum, e agendamento e visualização de consultas, no caso do usuário profissional, foram construídas e estão presentes nos MVPs. As principais telas serão apresentadas a seguir, em subseções contendo as interfaces em Thunkable e em Flutter. A demonstração de ambos MVPs foi registrada em um dispositivo Android tendo em vista que o da plataforma no-code não poderia gerar uma versão web sem custo.

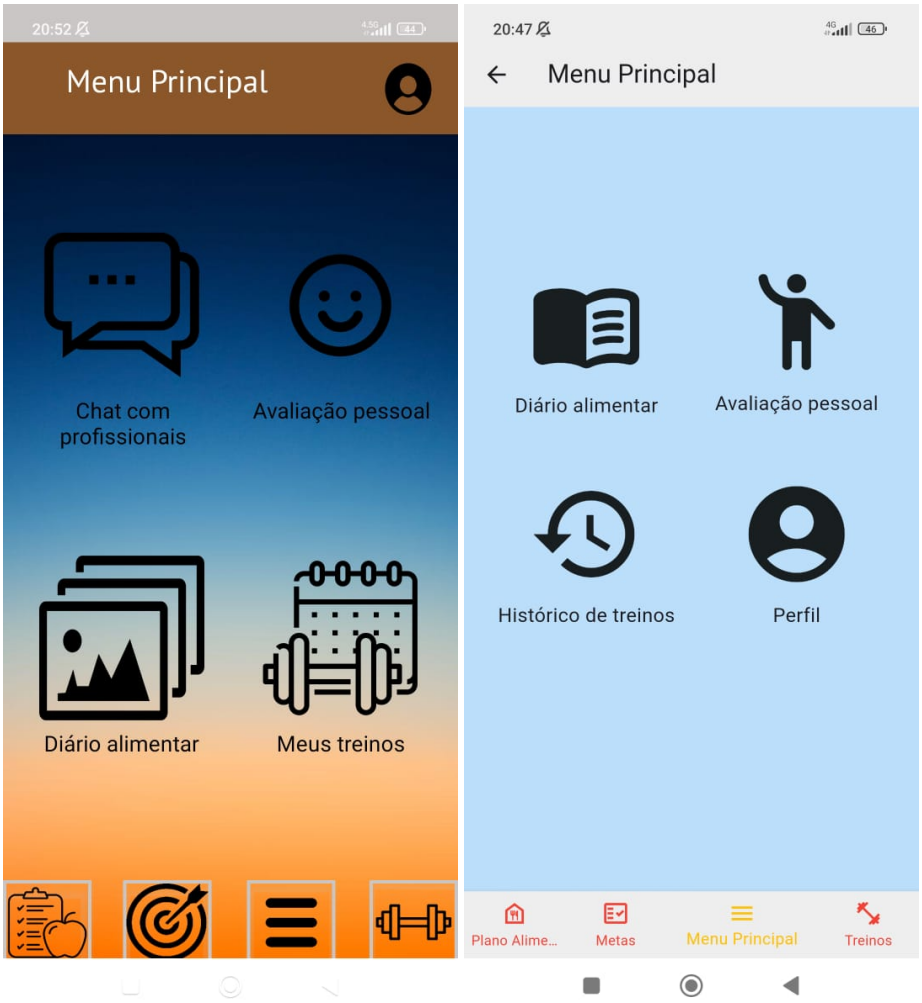
5.2 Funcionamento Usuário

Aqui serão apresentadas as telas do usuário comum.

5.2.1 Menu Principal

As Figuras 5.1a e 5.1b mostram as telas de menu principal do usuário comum.

Figura 5.1: Menu Principal



(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

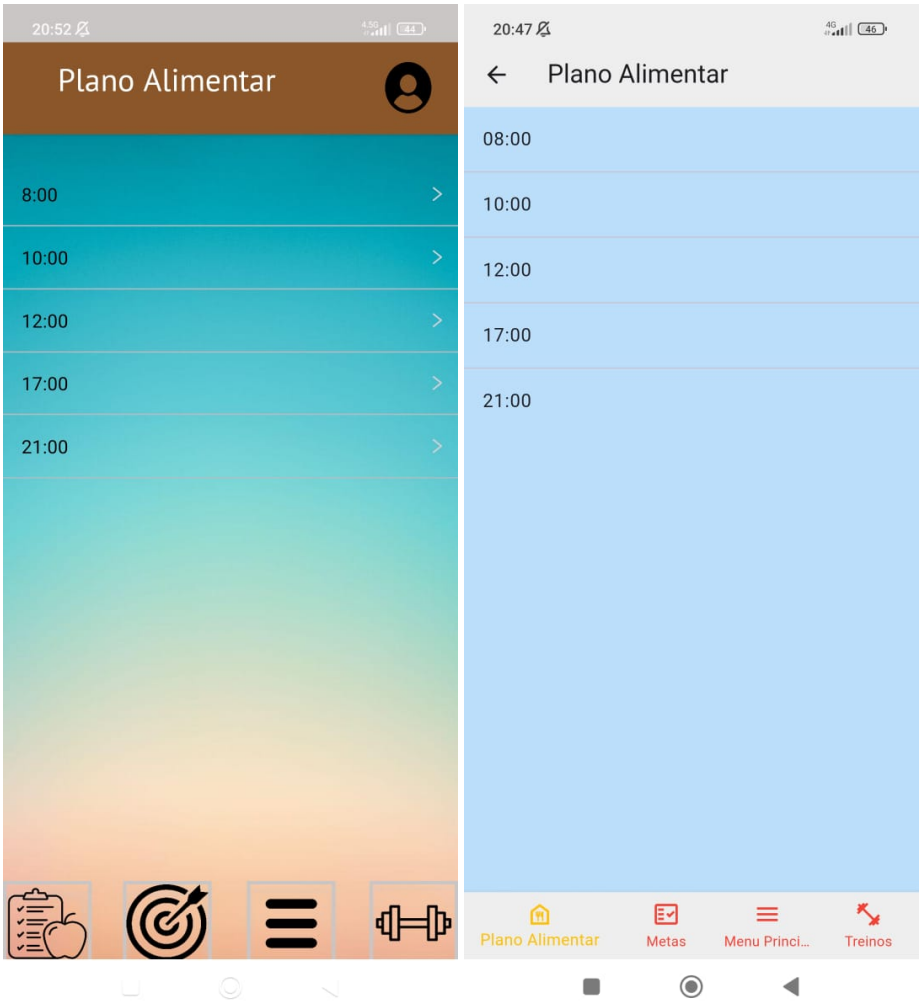
Como tela inicial do usuário, é apresentada a interface do menu principal, que serve como instrumento fundamental de navegação dentro do aplicativo - visto que é a única tela que pode acessar todas as funcionalidades. Alguns recursos mais utilizados, como plano alimentar, treinos e metas estão presentes em um navegador na base da página, sendo que esse navegador segue acessível quando o usuário seleciona uma dessas opções para ter um acesso rápido entre essas ferramentas.

Na tela da plataforma no-code podemos notar um campo para chat. Essa funcionalidade foi descartada para o MVP seguinte, visto que não era um requisito fundamental, e cedeu seu espaço à funcionalidade do perfil de usuário, que originalmente ficava no navegador do topo da página. Podemos notar também que os ícones da barra de navegação inferior receberam legenda com o nome da funcionalidade após feedback dos usuários no primeiro teste.

5.2.2 Plano Alimentar

As Figuras 5.2a e 5.2b mostram as telas dispondo os horários do plano alimentar do usuário comum.

Figura 5.2: Plano Alimentar



(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

A tela de plano alimentar tem como intuito apresentar uma lista de horários preestabelecidos pelo nutricionista para o usuário fazer suas refeições, sendo que cada um dos horários possui seu próprio cardápio.

As interfaces são essencialmente iguais, com a diferença da remoção da flecha ao final dos itens, visto que o acesso às refeições é intuitivo, e com a remoção do acesso ao perfil no navegador superior, já que ele foi movido para o menu principal.

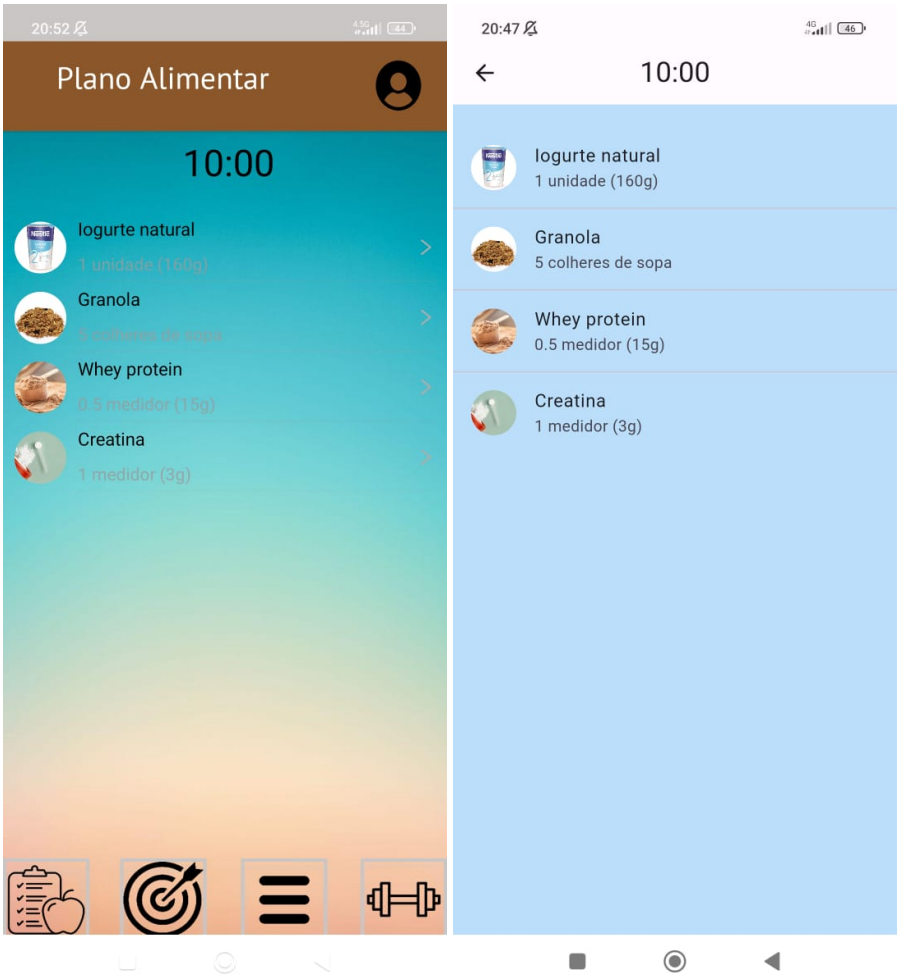
Apesar da tela de cardápio ser visualmente parecida, sua construção foi feita de

maneira diferente.

Focando primeiro nas alterações visuais, o texto contendo o horário passou a pertencer ao navegador superior, além da remoção do acesso ao perfil no mesmo navegador, já que ele foi movido para o menu principal. Além disso, o subtexto contendo as quantidades dos alimentos não pôde ter sua cor alterada, fato esse que gerou comentários sobre dificuldade de visualização entre usuários do primeiro MVP. Por último, a flecha na direita não pode ser removida de um componente *Data Viewer List* (nem seu símbolo pode ser alterado).

Focando na parte de *back-end*, a diferença mais importante se origina do fato de que o Thunkable não permite criar uma *Data Viewer List* capaz de alterar quais dados recebe, sendo necessário criar uma página para cada horário possível de ser selecionado (5 no total), - uma solução pouco elegante e sem escalabilidade. As duas telas podem ser observadas abaixo, nas Figuras 5.3a e 5.3b.

Figura 5.3: Cardápio



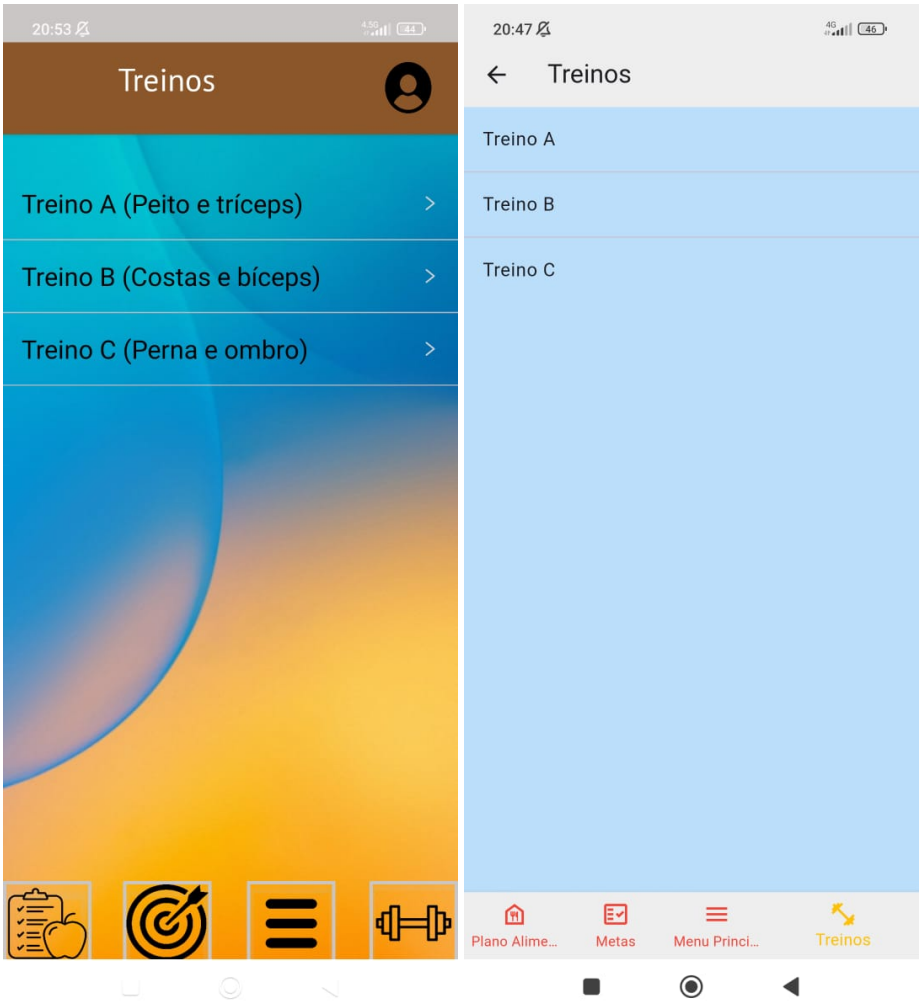
(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

5.2.3 Treinos

As Figuras 5.4a e 5.4b mostram as telas dispondo os treinos do usuário comum.

Figura 5.4: Treinos



(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

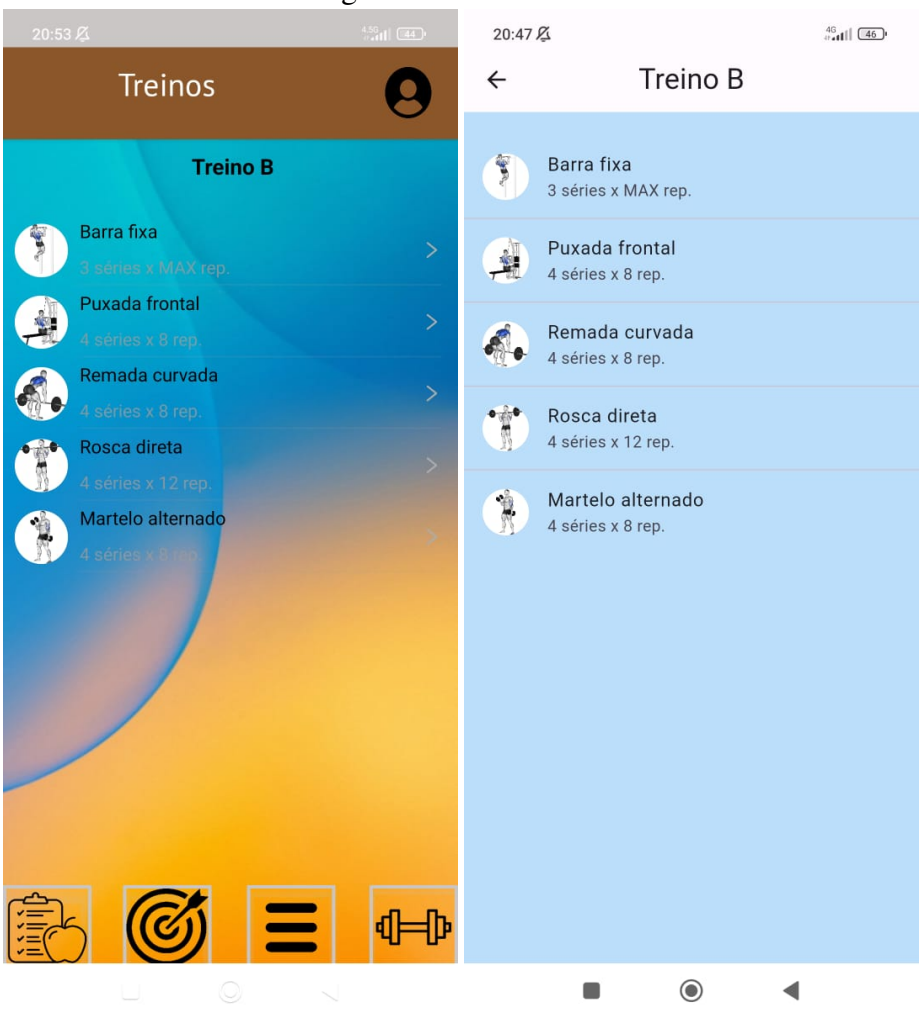
A tela de treinos tem como intuito apresentar uma lista de treinos preestabelecidos pelo educador físico para o usuário fazer seus exercícios de academia, sendo que cada um dos treinos possui sua própria rotina de atividades.

Novamente, as interfaces são essencialmente iguais, com as mesmas diferenças apontadas na tela de plano alimentar.

A tela dos exercícios segue o mesmo modelo de interface que é apresentado para a tela de cardápio, sendo observadas as mesmas diferenças entre os dois MVPs (com

três telas, uma para cada treino, ao invés de cinco). As duas telas podem ser observadas abaixo, na Figura 5.5.

Figura 5.5: Exercícios



(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

5.2.4 Metas

As Figuras 5.6a e 5.6b mostram as telas dispondo as metas do usuário comum.

Figura 5.6: Metas

(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

A tela de metas busca apresentar as metas acordadas entre os profissionais e o usuário de maneira a distinguir por faixa temporal cada conjunto de objetivos. Dito isso, é composta por três seções divididas em metas diárias, semanais e mensais - cada uma com um conjunto de lista de alvos a serem alcançados, seguidos de um botão de confirmação.

A funcionalidade de metas é a primeira que apresenta distinções notáveis entre os dois modelos de MVP. No caso da plataforma no-code, como foi utilizada um componente *Data Viewer List* para cada lista de metas, a funcionalidade de deslizar para acompanhar o restante dos objetivos ficou contida em cada lista - que não possibilita inserir botões no conjunto e também não possibilita que o clique na flecha à direita sirva como comando para ações. Tendo isso em vista, a única solução possível foi utilizar um comando nativo da classe que permite deslizar cada item para os lados (*Left swipe* nesse caso), possibilitando a inserção de uma confirmação à direita do item, como visto na Figura 5.7a.

No caso da plataforma tradicional, foi utilizada a classe *ScrollControler* para possibilitar o uso do deslizamento independente para cada lista de metas, que estavam contidas em uma classe *CustomScrollView*. Ademais, foi possível inserir botões de confirmação para cada objetivo utilizando a criação de um botão “Confirmar” - que é desabilitado após ser pressionado - para cada classe *Row* que contivesse uma meta, como visto na Figura 5.7b. Além disso, houve a remoção do acesso ao perfil no navegador superior, já que ele foi movido para o menu principal.

Figura 5.7: Confirmação de Metas

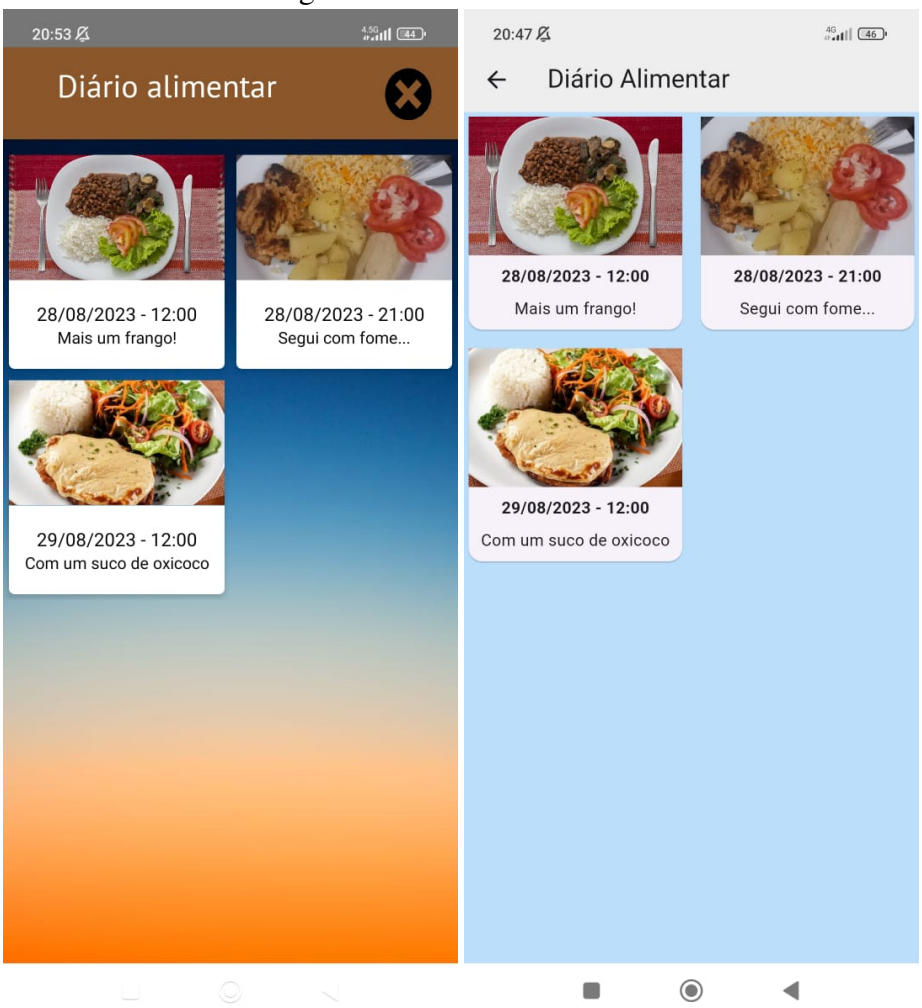
(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

5.2.5 Diário Alimentar

As Figuras 5.8a e 5.8b mostram as telas dispondo o diário alimentar do usuário comum.

Figura 5.8: Diário Alimentar



(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

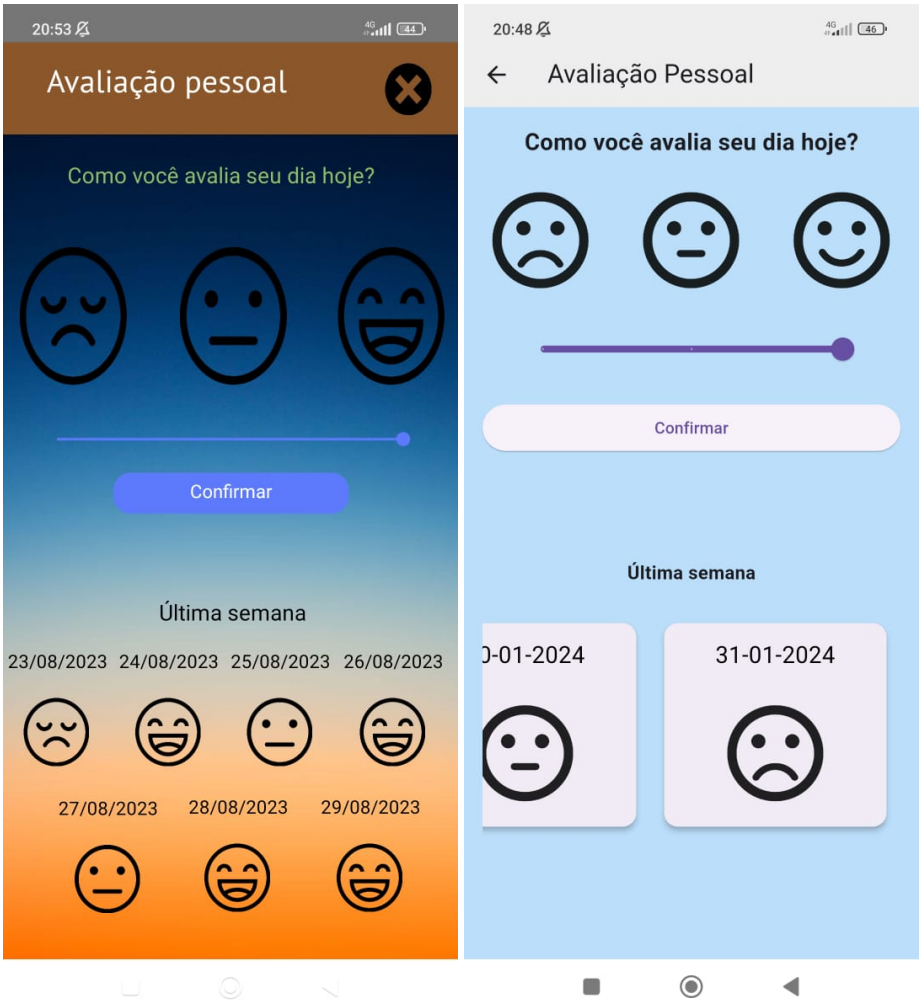
A tela de diário alimentar serve para o usuário acompanhar seus registros de alimentação, sendo que o MVP contém a visualização, mas não a inserção de novas entradas. Cada cartão representa uma refeição que o usuário comeu, contendo foto dos alimentos, data e comentário.

Ambos MVPs implementam essa funcionalidade de forma similar: no caso do Thunkable, é utilizado um componente *Data Viewer Grid*, com cada cartão composto por uma imagem seguida de dois textos; no caso do Flutter, é utilizada uma classe *Grid View* onde cada cartão é composto pelas mesmas partes que integram o cartão da plataforma no-code.

5.2.6 Avaliação Pessoal

As Figuras 5.9a e 5.9b mostram as telas dispondo a avaliação pessoal do usuário comum.

Figura 5.9: Avaliação Pessoal



(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

A tela de avaliação pessoal tem o propósito de permitir ao usuário avaliar seu próprio desempenho em relação a seus objetivos diariamente. Além da parte de inserção da avaliação na metade superior da tela, é apresentado um histórico de avaliações na parte inferior.

Visualmente falando, a maior diferença entre as duas telas é o histórico apresentando a última semana de desempenho do usuário, que na plataforma no-code é exibido com os sete últimos dias na tela, enquanto na plataforma tradicional é exibido através de uma classe *List View* que pode ser deslizada horizontalmente para mostrar as avaliações

registradas.

Falando no quesito *back-end*, entretanto, algumas diferenças a mais surgem, tal qual o uso de imagens para demonstrar o estado do usuário (emoji triste, emoji neutro e emoji feliz) no MVP feito em Thunkable, substituido pelo uso de ícones, da classe *Icon*, nativos do desenvolvimento em Flutter - removendo a necessidade de adicionar esses *assets* no projeto. Outra diferença é que a apresentação do histórico foi feita *hardcoded* em Thunkable por se tratar de um MVP, através da utilização de imagens e textos fixos, fato esse que seria considerado má prática de programação em um produto final - mas que foi contornado para o MVP em Flutter utilizando a data do dispositivo e enfileirando avaliações em uma lista com sete espaços.

Para o MVP na plataforma tradicional, a parte de confirmação realmente registra a avaliação, diferentemente do primeiro MVP, e a adiciona à lista do histórico, como pode ser observado abaixo na Figura 5.10.

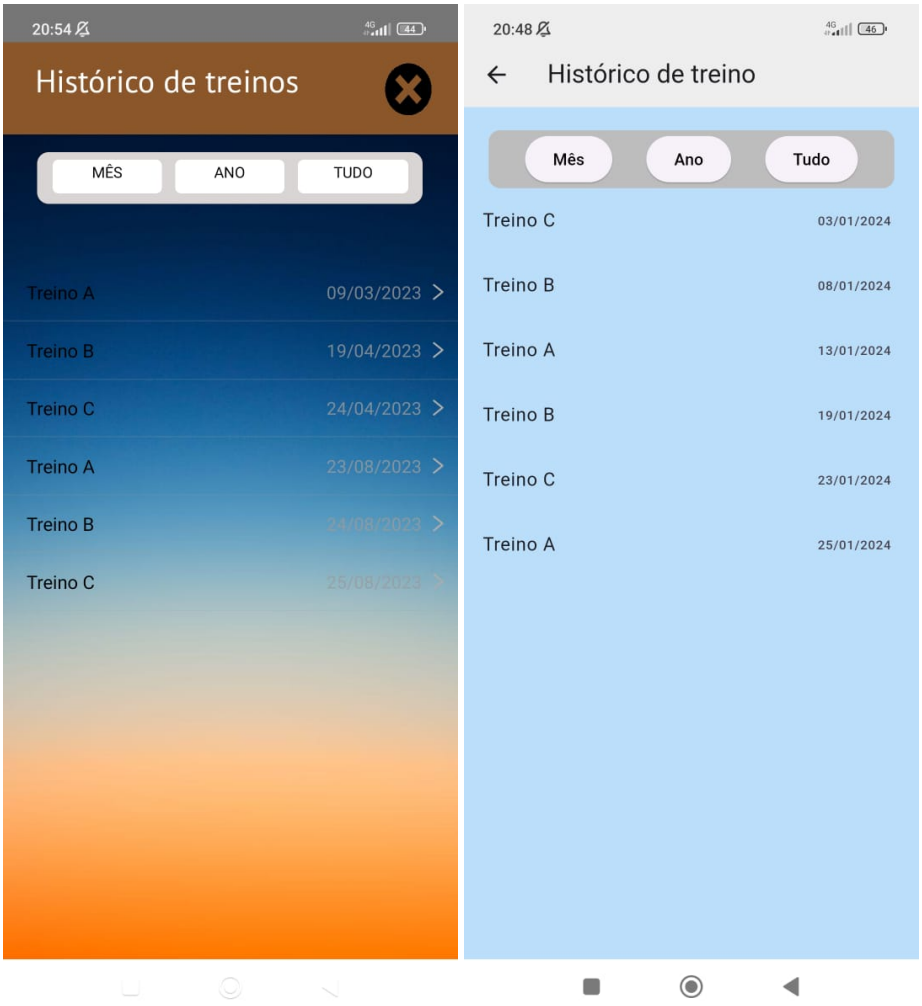
Figura 5.10: Confirmação de Avaliação Pessoal

Fonte: Gerado pelo autor

5.2.7 Histório de Treinos

As Figuras 5.11a e 5.11b mostram as telas dispondo o histórico de treinos do usuário comum.

Figura 5.11: Histórico de Treinos



(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

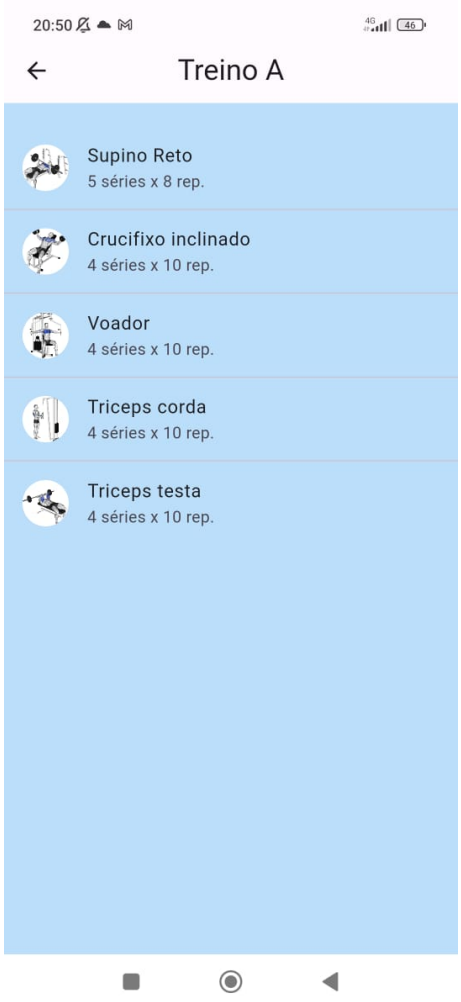
A tela de histórico de treinos tem como intuito fornecer ao usuário um registro de seus treinos, podendo diferenciar por mês, ano ou histórico completo.

Virtualmente iguais, aqui ocorre novamente o problema de não poder retirar a flecha à direita de cada item em Thunkable, dando a entender que a escolha pode ser acessada - o que não é verdade. Ambas as interfaces possuem um container composto por três botões na parte superior da tela, seguido por uma lista de treinos. É importante comentar que em Thunkable, como um componente *Data Viewer List* só pode ter uma fonte de dados, foram construídos três componentes iguais - um para cada opção - e todos têm

sua visibilidade definida como “falso”, alterando a visibilidade de cada lista conforme o clique dos botões presentes no topo da tela.

Para o MVP na plataforma tradicional, a parte do histórico é mais completa e apresenta os exercícios realizados no dia, como pode ser observado abaixo na Figura 5.12, *feature* essa que não está disponível no MVP da plataforma no-code.

Figura 5.12: Exercícios do Histórico de Treinos

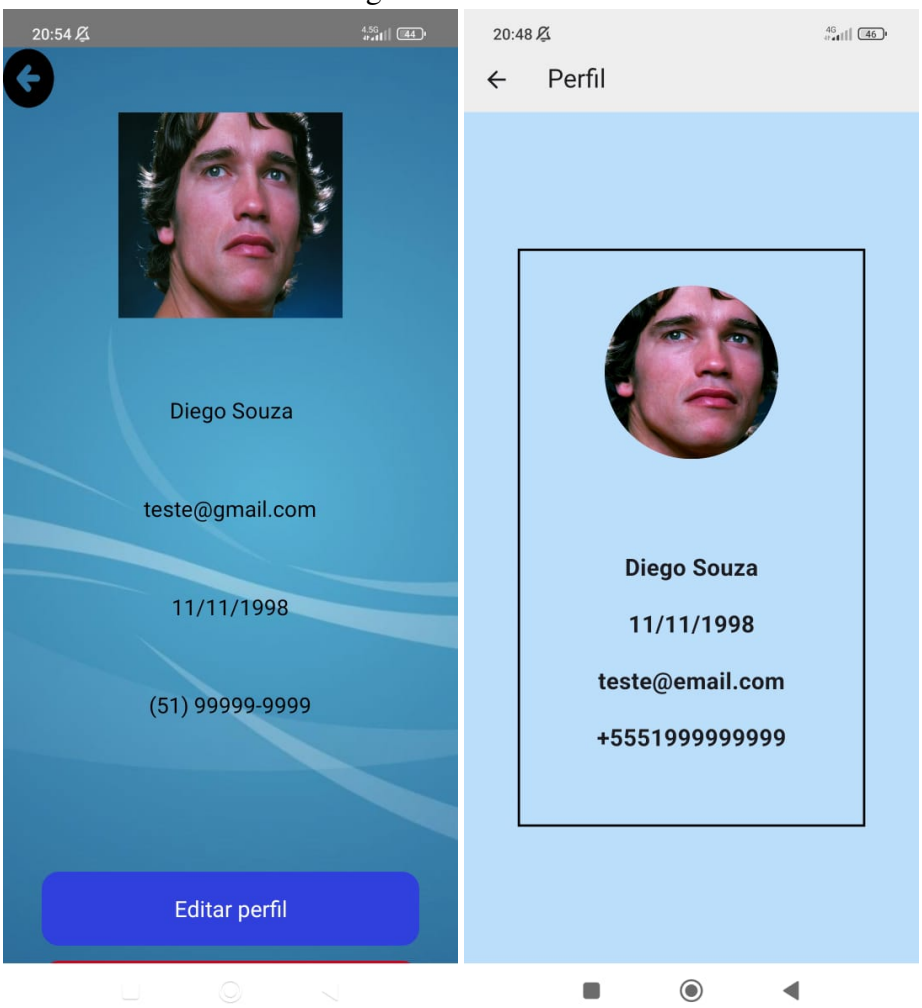


Fonte: Gerado pelo autor

5.2.8 Perfil

As Figuras 5.13a e 5.13b mostram as telas de perfil do usuário comum.

Figura 5.13: Perfil



(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

A tela de perfil tem como objetivo apresentar os dados do usuário que está utilizando a plataforma.

Algumas diferenças notáveis são os botões de editar perfil e sair na parte de baixo da interface - note que o botão sair quase não aparece, ainda que na função *Web Preview* ele esteja inteiramente dentro da tela, um erro da plataforma ao passar para o dispositivo - que foram considerados desnecessários para o escopo de um MVP. Outra diferença é a imagem, que na plataforma no-code não pode ter seu formato alterado, enquanto na plataforma tradicional foi englobada por uma classe “CircleAvatar” que permite anexar a imagem em um formato circular. Por último, nota-se uma moldura no caso do Flutter, feita facilmente por uma classe “Container” - moldura essa que foi evitada em Thunkable por motivos já discutidos sobre o uso do “Canvas”.

5.3 Funcionamento Profissional

Aqui serão apresentadas as telas do usuário profissional. É importante lembrar que a parte de alteração em planos alimentares e treinos não entrou no escopo do MVP e, por conseguinte, não está aqui representada.

5.3.1 Pacientes

As Figuras 5.14a e 5.14b mostram as telas dispondo os pacientes do usuário profissional.

Figura 5.14: Pacientes

(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

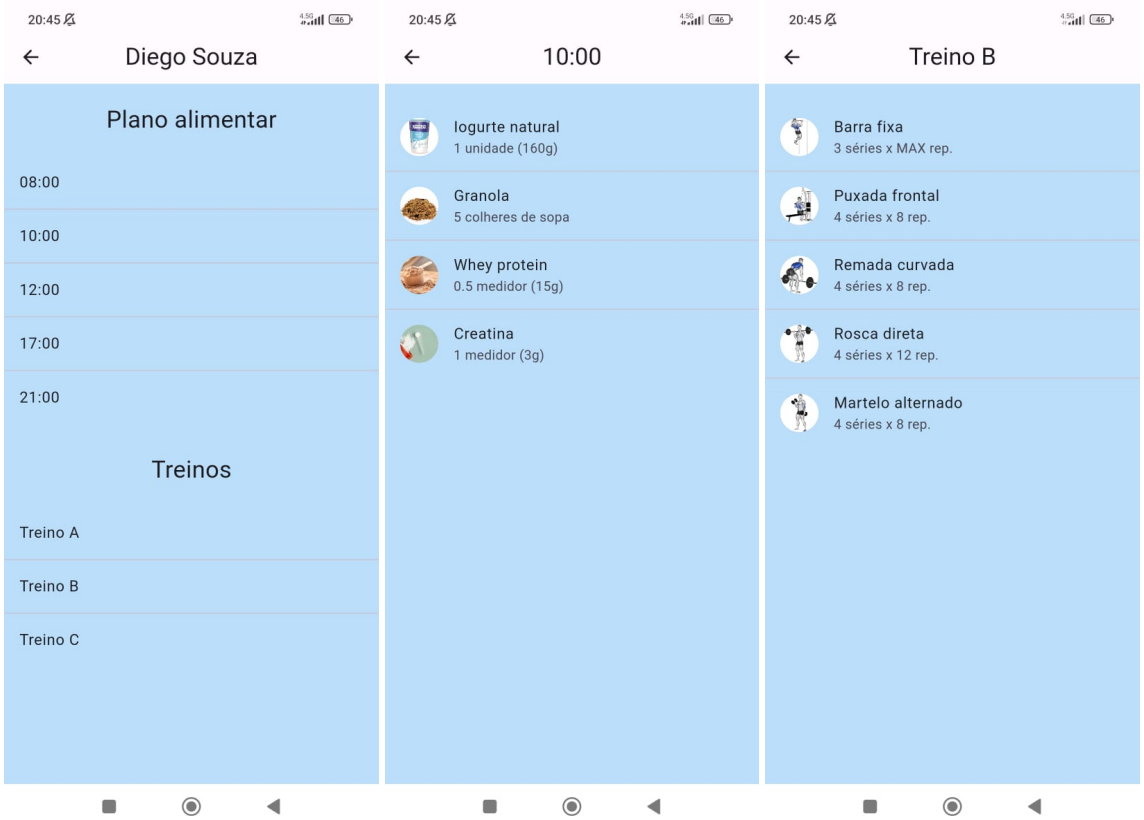
Apresentada como tela inicial do profissional, a interface dos pacientes exibe uma

lista de clientes, acompanhada de um navegador na base da página contendo as outras funcionalidades, como calendário, criação de consultas e perfil - sendo que esse navegador segue acessível quando o usuário seleciona uma dessas opções.

Na tela da plataforma no-code podemos notar um campo para chat. Como já comentado na parte do usuário, essa funcionalidade foi descartada para o MVP seguinte, visto que não era um requisito fundamental, e cedeu seu espaço à funcionalidade de marcação de consulta, que originalmente ficava no clique de um paciente da lista. Podemos notar também que os ícones da barra de navegação inferior receberam legenda com o nome da funcionalidade após feedback dos usuários no primeiro teste.

Com a exclusão do chat, a ação de seleção de um paciente ficou com espaço vago. Para ocupar esse espaço, foi implementada a *feature* de visualizar o plano alimentar e treino de um paciente, tal qual exibido na Figura 5.15a - podendo selecionar um horário do plano alimentar, caso em que a tela exibida será a da Figura 5.15b, ou um treino, caso em que a tela exibida será a da Figura 5.15c.

Figura 5.15: Paciente



(a) Exibição da tela de um paciente (b) Exibição de uma refeição do paciente (c) Exibição de um treino do paciente

Fonte: Gerado pelo autor

5.3.2 Calendário

As Figuras 5.16a e 5.16b mostram as telas dispondo calendário e agendamentos do usuário profissional.

Figura 5.16: Calendário

(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

A tela de calendário busca apresentar ao profissional os seus agendamentos de consultas. Ainda que o calendário presente na tela não esteja servindo para a função de apresentar eventos, foi escolhido mantê-lo para exibir como ficaria em uma versão final. Após o calendário é exibida uma lista de agendamentos com o nome do cliente, a data e horário.

Para a tela da plataforma no-code, foi utilizado um componente *Web Viewer* apontando para a URL do calendário da Google, o que gerou problemas no teste pois os usuá-

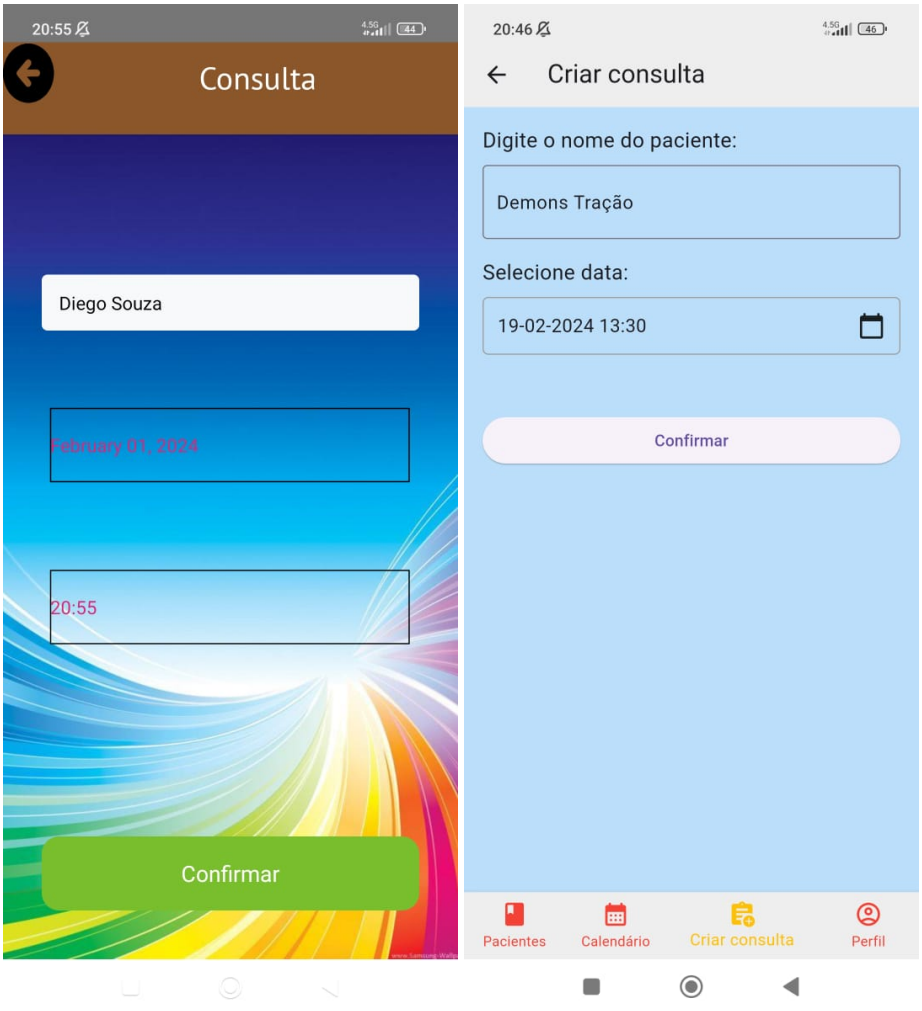
rios precisariam entrar com sua própria conta. Além disso, o tamanho do calendário foi alterado entre a função *Web Preview* e o aplicativo no dispositivo, o que gerou sobreposição do texto “Agendamentos”. A cor do subtexto do componente *Data Viewer List* novamente atrapalhou a visualização dos testadores.

Para a tela da plataforma tradicional, foi utilizada a classe *TableCalendar* pertencente à biblioteca homônima *TableCalendar* para apresentar o calendário.

5.3.3 Criar Consulta

As Figuras 5.17a e 5.17b mostram as telas de criar consulta do usuário profissional.

Figura 5.17: Criar Consulta



(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

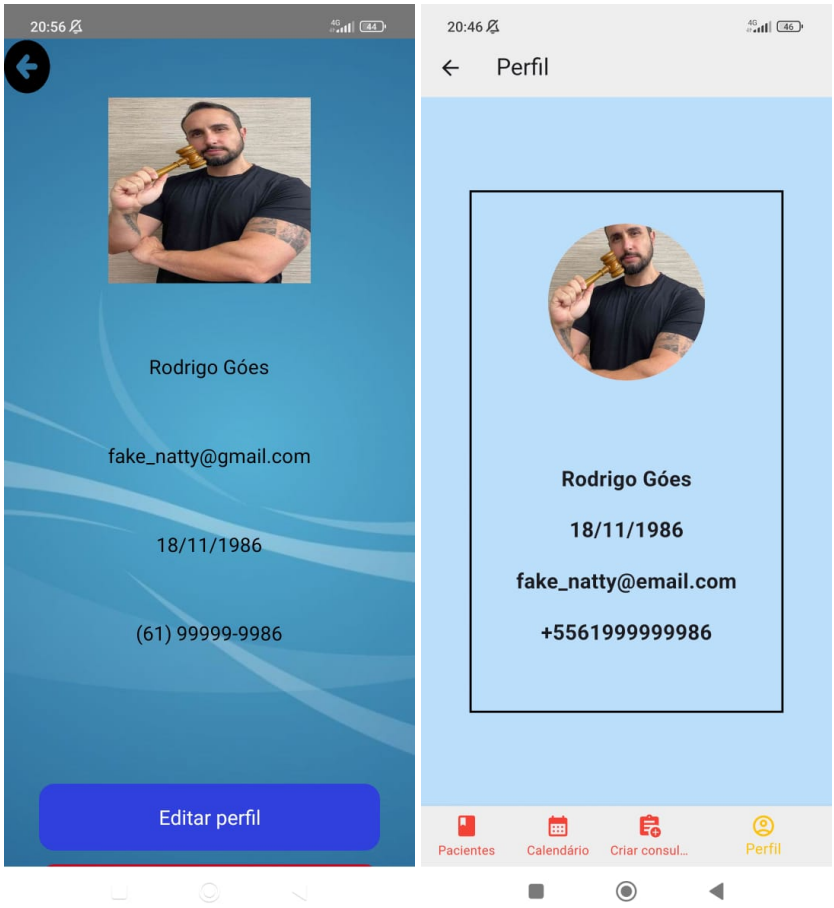
A tela de criação de consultas busca disponibilizar ao profissional um método para inserção de novos agendamentos. A tela é composta de um *input* de texto onde deve ser preenchido o nome do paciente, seguido da seleção de data e hora e de um botão de confirmação.

Na tela da plataforma no-code podemos ver que a seleção de data e hora estão separados entre si. O motivo disso é que em Thunkable não há um componente que possua as duas entradas, sendo necessário um componente *Date Input* e um *Time Input*. Na plataforma tradicional, entretanto, é apenas uma seleção, utilizando a classe nativa de Flutter *Inkwell* com uma função criada *_pickDateTime* que utiliza as funções *showDatePicker* e *showTimePicker*.

5.3.4 Perfil

As Figuras 5.18a e 5.18b mostram as telas de perfil do usuário profissional.

Figura 5.18: Perfil



(a) Thunkable (b) Flutter

Fonte: Gerado pelo autor

A tela de perfil tem como objetivo apresentar os dados do profissional que está utilizando a plataforma.

As mudanças do segundo MVP para o primeiro seguem as mesmas alterações que foram feitas para o perfil do usuário comum, sendo que a única diferença é que na plataforma tradicional foi adicionado um navegador na base da interface por não ter um menu principal.

5.4 Limitações

Como o enfoque desse trabalho era a criação de MVPs, algumas decisões foram tomadas durante o desenvolvimento para seguir no escopo estabelecido dentro do prazo de entrega, deixando algumas funcionalidades para a sequência do aplicativo e limitando algumas outras funcionalidades que já estão presentes no MVP.

A *feature* mais clara, e talvez mais grave, que ficou de fora do trabalho é a adição de novos pacientes no aplicativo do profissional. Outras *features* como a inserção de treinos para aparecerem no "Histórico de Treinos", a inserção de entradas no "Diário Alimentar" e a visualização de agendamentos por parte do usuário comum são menos importantes, mas também seriam desejáveis de se dispor.

Já dentre as features que estão no MVP, mas não ficaram completas, podemos notar o calendário do profissional que funciona, mas não exibe eventos. Podemos notar também que no agendamento é necessário digitar o nome do paciente, o que poderia ser substituído por uma *drop-down list* com possibilidade de digitação para filtrar em tempo real. Ainda, no caso do usuário comum, não foi definido por quanto tempo o botão de metas ficará desabilitado após ser pressionado, nem foi estabelecido o controle de ter apenas uma avaliação diária. Por último, no caso do profissional, a organização da lista de agendamentos não ordena por data, e sim por inserção.

6 EXPERIMENTOS E AVALIAÇÃO

Este capítulo tem como objetivo descrever os teste de usabilidade que foram realizados com desenvolvedores e usuários.

6.1 Visão Geral

O teste da aplicação foi dividido em duas partes, uma avaliação preditiva através de uma avaliação heurística com programadores seguida de uma avaliação formativa através de um teste com usuários, sendo que ambos os testes foram aplicados tanto após a criação do MVP em Thunkable, quanto em Flutter.

A avaliação heurística foi realizada com cinco desenvolvedores buscando encontrar erros que pudessem afetar o funcionamento do aplicativo - visto que promove um *feedback* rápido - sendo utilizadas as heurísticas de Nielsen, um método famoso de inspeção de usabilidade criado por Jakob Nielsen e Rolf Molich na década de 90. O método busca avaliar se os elementos de uma interface seguem os princípios testados e aprovados, já padronizados na indústria.

O teste da aplicação foi realizado com 12 participantes para a plataforma no-code e 20 para a plataforma tradicional - incluindo 9 participantes que estiveram em ambos os testes -, sendo que o maior número de participantes no segundo caso se dá por conta do Flutter permitir a criação de aplicativos compilados nativamente para web, enquanto o modelo de negócios do Thunkable só permite a criação de aplicação web a partir do *pro plan* ($45/mês). O objetivo foi conseguir avaliar o aplicativo com uma demografia diversificada, permitindo validar o uso para maior quantidade de usuários. O teste foi composto por tarefas seguidas de um questionários ASQ para cada - questionário esse criado por Jim Lewis em 1995 para avaliar a satisfação do usuário de forma rápida, contendo três perguntas sobre a tarefa realizada - seguidas de um questionário SUS, método criado por John Brooke em 1986 para avaliar a eficiência, efetividade e satisfação do usuário em 10 perguntas sobre a usabilidade do sistema como um todo. Em ambos os questionários foi aplicada a escala Likert (LIKERT, 1932) para as respostas, com 7 pontos no ASQ e 5 pontos no SUS, avaliando o quão fortemente o usuário concorda ou discorda com cada afirmação.

6.2 Protocolo

Para a avaliação heurística, foi pedido para que desenvolvedores explorassem a aplicação sistematicamente, buscando cobrir a maior parte das funcionalidades. Após o uso, lhes foi pedido que preenchessem um documento (Apêndice A) que serviria para registrar os problemas encontrados e sua severidade. Para garantir que os testadores estivessem a par do significado das heurísticas utilizadas, foram fornecidas descrições de apoio para cada heurística e exemplos de preenchimento.

Para ser realizada a avaliação dos usuários, foi pedido para que os testadores realizassem duas tarefas preestabelecidas - a primeira simulando o uso de um usuário comum e a segunda simulando o uso de um profissional - e cronometrassem o tempo demandado para completar cada tarefa. Dentre as atividades realizadas pelo usuário comum, estavam afazeres como: visualização da dieta, visualização dos treinos, confirmação de metas; dentre as atividades realizadas pelo usuário profissional, estavam afazeres como: visualização dos pacientes, marcação de consulta, visualização de consultas agendadas. As tarefas estavam em um formulário online a ser preenchido (Apêndice B) que era dividido em informações demográficas, tarefas com o questionário ASQ na sequência, e o questionário SUS ao final para avaliar o uso geral da aplicação.

6.3 Participantes

Para os participantes da avaliação preditiva foram convidados 5 programadores, ou formados em Ciência da Computação pela UFRGS, ou ainda cursando essa mesma graduação, buscando indivíduos com melhor entendimento do processo para prover uma análise mais detalhada. Ainda, para aumentar a garantia de erros encontrados - visto que os 2 MVPs gerados são similares - 3 dos desenvolvedores foram alterados entre a avaliação da plataforma no-code e a avaliação da plataforma tradicional.

Para os participantes da avaliação formativa, foram convidados participantes de idades e gêneros diferentes, buscando um teste que abrangesse diversos públicos. Para o MVP em Thunkable foram convidados 12 participantes que precisaram testar em dispositivos Android baixando um arquivo executável .apk, enquanto para o MVP em Flutter foram convidados 20 participantes que poderiam optar por testar também baixando um arquivo executável .apk para dispositivos Android, ou acessando um link para a aplicação web - podendo ser testado em qualquer navegador, desde que em dispositivos com sistema

operacional Android, Linux ou Windows.

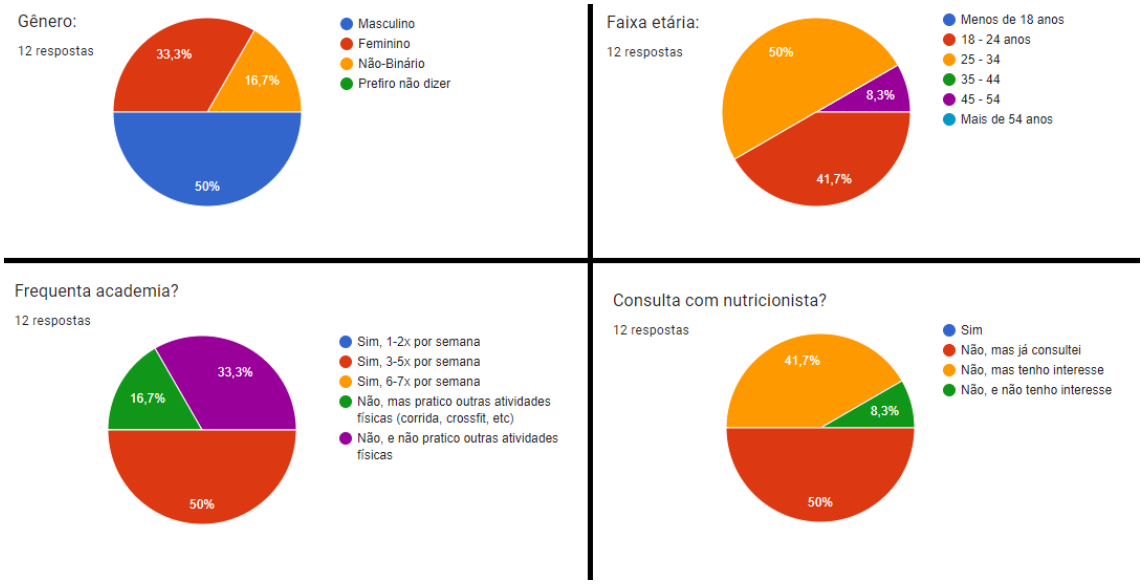
Os resultados distintos estão apresentados nas seções abaixo, com uma comparação na sequência.

6.4 Thunkable

O questionário foi respondido por 12 usuários com idade e gênero distintos, tendo enfoque nas perguntas direcionadas para validar que a aplicação teria público com interesse na sua solução. Metade dos usuários afirmou que frequenta academia, enquanto 11 dos 12 participantes afirmaram que, ou já frequentaram nutricionista, ou têm o interesse em frequentar. Esse resultado demonstra um grande público possível de usuários para a aplicação.

Os resultados estatísticos sobre as informações demográficas se encontram na Figura 6.1.

Figura 6.1: Estatísticas Demográficas Thunkable



Fonte: Gerado pelo autor

Utilizando os resultados do questionário ASQ, podemos calcular uma nota para o sentimento do usuário em relação a cada tarefa realizando a média aritmética das 3 questões. A nota obrigatoriamente ficará entre 1 e 7 que são os extremos da escala, com 1 representando a pior nota possível e 7 representando a melhor nota possível. Dito isso, podemos encontrar que a nota para a primeira tarefa ficou 6.33, enquanto para a segunda

tarefa ficou 6.36. Esse resultado comprova que os usuários ficaram satisfeitos com o uso do aplicativo como ferramenta para completar as tarefas.

O resultado do questionário ASQ encontra-se na Tabela 6.1.

Tabela 6.1: Resultados do Questionário ASQ - Thunkable

Tarefa 1				
Questão	Enunciado	Concorda	Neutro	Discorda
Q1	Estou satisfeito quanto a facilidade para completar a tarefa.	100%	0%	0%
Q2	Estou satisfeito quanto ao tempo que levou para completar a tarefa.	100%	0%	0%
Q3	Estou satisfeito com o nível de suporte e informação enquanto completava a tarefa.	91.6%	8.4%	0%
Tarefa 2				
Questão	Enunciado	Concorda	Neutro	Discorda
Q1	Estou satisfeito quanto a facilidade para completar a tarefa.	91.6%	0%	8.4%
Q2	Estou satisfeito quanto ao tempo que levou para completar a tarefa.	91.6%	8.4%	0%
Q3	Estou satisfeito com o nível de suporte e informação enquanto completava a tarefa.	91.6%	8.4%	0%

Fonte: O Autor

Utilizando os resultados do questionário SUS, podemos calcular uma nota para o sentimento do usuário em relação à aplicação como um todo, utilizando a fórmula de Brooke[1] para o cálculo: para respostas em questões ímpares, subtrair 1 da escolha do usuário; para respostas em questões pares, subtrair a resposta de 5; somar os valores e multiplicar por 2.5. Esse teste resulta em um número entre 0 e 100, sendo que 0 representa o pior dos casos e 100 representa o melhor - e, nesse caso, o teste obteve 85.2 de pontuação, resul-

[1]BROOKE, John et al. SUS-A quick and dirty usability scale. Usability evaluation in industry, v. 189, n. 194, p. 4-7, 1996.

tado que se encontra na classifação A (acima de 80.3), considerada a melhor classificação possível[2].

O resultado do questionário SUS encontra-se na Tabela 6.2.

Tabela 6.2: Resultados do Questionário SUS - Thunkable

Questão	Enunciado	Concorda	Neutro	Discorda
Q1	Eu acho que gostaria de usar esse sistema com frequência.	75%	25%	0%
Q2	Eu acho o sistema desnecessariamente complexo.	0%	0%	100%
Q3	Eu achei o sistema fácil de usar.	91.6%	0%	8.4%
Q4	Eu acho que precisaria de ajuda de uma pessoa com conhecimentos técnicos para usar o sistema.	8.4%	0%	91.6%
Q5	Eu acho que as várias funções do sistema estão muito bem integradas.	83.3%	16.7%	0%
Q6	Eu acho que o sistema apresenta muita inconsistência.	0%	8.4%	91.6%
Q7	Eu imagino que as pessoas aprenderão como usar esse sistema rapidamente.	91.6%	0%	8.4%
Q8	Eu achei o sistema atrapalhado de usar.	0%	8.4%	91.6%
Q9	Eu me senti confiante ao usar o sistema.	91.6%	0%	8.4%
Q10	Eu precisei aprender várias coisas novas antes de conseguir usar o sistema.	0%	0%	100%

Fonte: O Autor

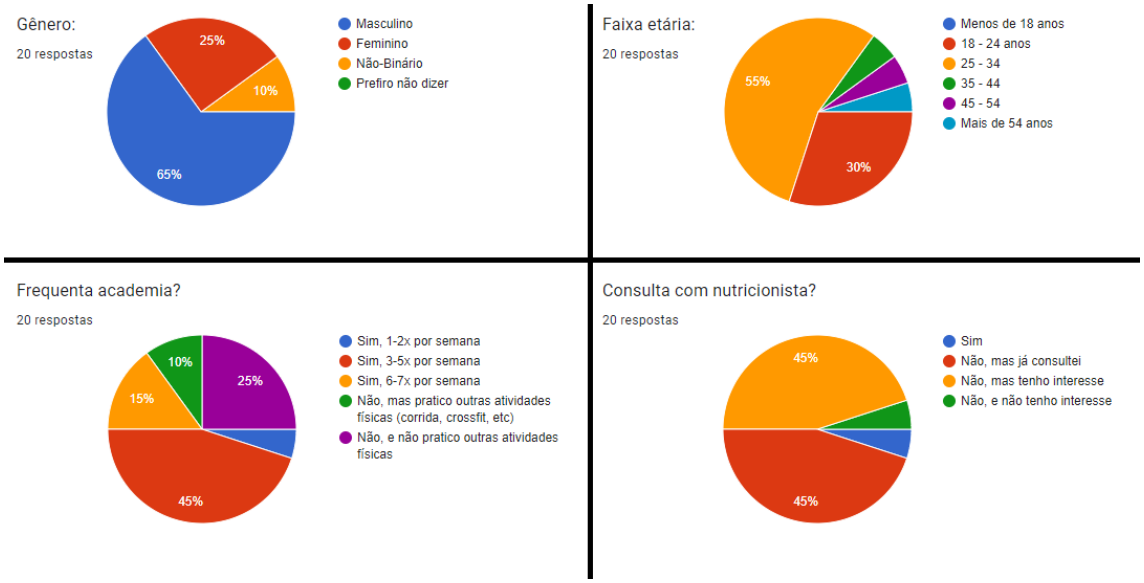
[2]https://uiuxtrend.com/measuring-system-usability-scale-sus/ (acesso em fevereiro de 2024)

6.5 Flutter

O questionário foi respondido por 20 usuários com idade e gênero distintos, tendo enfoque nas perguntas direcionadas para validar que a aplicação teria público com interesse na sua solução. 65% dos usuários afirmaram que frequentam academia, enquanto 95% participantes afirmaram que frequentam nutricionista, ou já frequentaram, ou têm o interesse em frequentar. Esse resultado confirma e reforça o que já havia sido percebido no teste da plataforma no-code.

Os resultados estatísticos sobre as informações demográficas se encontram na Figura 6.2.

Figura 6.2: Estatísticas Demográficas Flutter



Fonte: Gerado pelo autor

Tal qual no MVP da plataforma no-code, utilizando os resultados do questionário ASQ, podemos calcular uma nota para o sentimento do usuário em relação a cada tarefa realizando a média aritmética das 3 questões. Dessa vez, encontramos que a nota para a primeira tarefa ficou 6.38, enquanto para a segunda tarefa ficou 6.35.

O resultado do questionário ASQ encontra-se na Tabela 6.3.

Tabela 6.3: Resultados do Questionário ASQ - Flutter

Tarefa 1				
Questão	Enunciado	Concorda	Neutro	Discorda
Q1	Estou satisfeito quanto a facilidade para completar a tarefa.	95%	0%	5%
Q2	Estou satisfeito quanto ao tempo que levou para completar a tarefa.	100%	0%	0%
Q3	Estou satisfeito com o nível de suporte e informação enquanto completava a tarefa.	90%	10%	0%
Tarefa 2				
Questão	Enunciado	Concorda	Neutro	Discorda
Q1	Estou satisfeito quanto a facilidade para completar a tarefa.	100%	0%	0%
Q2	Estou satisfeito quanto ao tempo que levou para completar a tarefa.	100%	0%	0%
Q3	Estou satisfeito com o nível de suporte e informação enquanto completava a tarefa.	95%	5%	0%

Fonte: O Autor

Mais uma vez utilizando os resultados do questionário SUS, podemos calcular uma nota para o sentimento do usuário em relação à aplicação como um todo, utilizando novamente a fórmula de Brooke para o cálculo. Esse segundo teste obteve 87.1 de pontuação, resultado que também se encontra na classificação A (acima de 80.3).

O resultado do questionário SUS encontra-se na Tabela 6.4.

Tabela 6.4: Resultados do Questionário SUS - Flutter

Questão	Enunciado	Concorda	Neutro	Discorda
Q1	Eu acho que gostaria de usar esse sistema com frequência.	90%	10%	0%
Q2	Eu acho o sistema desnecessariamente complexo.	0%	10%	90%
Q3	Eu achei o sistema fácil de usar.	100%	0%	0%
Q4	Eu acho que precisaria de ajuda de uma pessoa com conhecimentos técnicos para usar o sistema.	0%	0%	100%
Q5	Eu acho que as várias funções do sistema estão muito bem integradas.	90%	5%	5%
Q6	Eu acho que o sistema apresenta muita inconsistência.	0%	15%	85%
Q7	Eu imagino que as pessoas aprenderão como usar esse sistema rapidamente.	95%	5%	0%
Q8	Eu achei o sistema atrapalhado de usar.	5%	5%	90%
Q9	Eu me senti confiante ao usar o sistema.	95%	5%	0%
Q10	Eu precisei aprender várias coisas novas antes de conseguir usar o sistema.	0%	10%	90%

Fonte: O Autor

6.6 Comparação

Nessa seção, ASQ1 e ASQ2 representarão os questionários ASQ das tarefas 1 e 2 respectivamente, enquanto SUS representará o questionário SUS.

Ambos os testes permitiram comprovar que, em geral, os usuários ficaram satisfeitos com a aplicação, tanto para a realização das tarefas, quanto para seu uso como um todo e, isso posto, é observável que o uso de Flutter teve mais sucesso ao gerar um MVP do que o Thunkable. Ainda que seja uma margem de diferença pequena, como veremos na tabela abaixo, precisa ser levado em conta que são 2 MVPs de uma mesma abstração, então qualquer disparidade deveria ser considerada relevante.

Os resultados da comparação encontram-se na Tabela 6.5.

Tabela 6.5: Comparação geral

Plataforma	ASQ1	ASQ2	SUS
No-code	6.33	6.36	85.2
Tradicional	6.38	6.35	87.1

Fonte: O Autor

É importante notar, ainda, a diferença entre as notas dos usuários que testaram os 2 MVPs (9 usuários), que ressalta mais ainda o argumento feito sobre a disparidade entre os resultados. Não só houveram alguns comentários no *feedback* opcional ao final do questionário sobre evolução notável do aplicativo, como a nota do questionário SUS somente desses usuários passou de 84.2 na plataforma no-code para 88.3 na plataforma tradicional, um ganho mais de duas vezes maior do que o ganho observando usuários em geral. É de suma importância pontuar, entretanto, que da visão de desenvolvedor não há "evolução" entre um MVP e outro pois são dois aplicativos construidos de forma completamente independente, ponto esse que passou despercebido aos usuários que testaram a aplicação.

7 CONCLUSÃO

Este trabalho teve como intenção mostrar que há um espaço aberto no mercado para uma aplicação integrada entre treinos e planos alimentares e também comparar a criação de um MVP para essa aplicação através de duas abordagens distintas. Os aplicativos cumpriram os objetivos estimados nos requisitos, tanto para o usuário comum, quanto para o usuário profissional. Nos próximos meses, é previsto que a criação do aplicativo seja continuada, dessa vez com enfoque na criação da aplicação final. Os resultados dos experimentos apresentados demonstram não só que o público-alvo existe e é grande, como também que os MVPs protótipos criados foram satisfatórios para seus propósitos.

Considerando o tempo e quantidade de esforço demandado para cada MVP, o autor propõe que não seja feita necessariamente uma escolha unificada por uma plataforma, e sim que seja criada uma versão inicial em no-code antes da criação do MVP em linguagem tradicional - por ser mais rápido, mais barato e por validar alguns pontos de antemão. É costumeiro que um produto tenha múltiplos MVPs ao longo do seu desenvolvimento, então nada impede que os iniciais sejam em uma estrutura diferente dos finais, e isso contornaria as limitações que as plataformas no-code apresentam, já validando alguns pontos do produto/ideia.

7.1 Trabalhos Futuros

Tendo em vista que foi construído um MVP, o próximo passo seria utilizar o *feedback* dos testadores para fazer algumas alterações e construir o aplicativo completo. Algumas das possibilidades futuras seriam:

- integração para gerir pacientes e seus treinos/dietas (web ou no próprio aplicativo);
- login através do Firebase Authentication, com integração de conta com Google/Facebook;
- expandir relatório de avaliação pessoal para mais de uma semana;
- aba de evolução física para o usuário com dados históricos como índice de gordura corporal, quantidade de massa magra, etc; e
- exibir mais de um cardápio por horário, caso o nutricionista dê mais de uma opção de refeição.

Além disso, é necessário realizar um estudo sobre design de aplicativo, tendo em vista que houveram comentários de estética feia no MVP no-code e que foi feita uma decisão de design simples no MVP tradicional, mas que não é o ideal. Outra possibilidade é expandir o uso do aplicativo para utilizar apenas um profissional.

8 APÊNDICES

8.1 Apêndice A - Avaliação aplicada a desenvolvedores

Neste apêndice encontra-se a avaliação heurística examinada por desenvolvedores.

Relatório de Avaliação Heurística

- Informações Complementares:
  Teste realizado no dispositivo individual de cada testador. Especificações:
  - Modelo:
  - RAM:
  - Espaço de armazenamento máximo:
  - CPU:
  - Tela:
  - Resolução:
  - Sistema:
  - Interface:
  - Modelo:
- As heurísticas a serem utilizadas na avaliação de usabilidade são:
  1. Visibilidade do status do sistema;
  2. Compatibilidade do sistema com o mundo real;
  3. Controle do usuário e liberdade;
  4. Consistência e padrões;
  5. Prevenção de erros;
  6. Reconhecimento ao invés de relembrança;
  7. Flexibilidade e eficiência de uso;
  8. Estética e design minimalista;
  9. Ajudar os usuários a reconhecer, diagnosticar e corrigir erros;
  10. Help e documentação.

- Classificação de severidade:

Grau	Importância	Consequência
0	Sem importância	Não afeta a operação
1	Cosmético	Não há necessidade imediata de reparo
2	Simples	Baixa prioridade para reparo
3	Grave	Alta prioridade para reparo
4	Catastrófico	Prioridade máxima para reparo

- Tabela de violações de heurísticas encontradas pelo avaliador:

N.	Problema	Heurística(s) Violada(s)	Severidade
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			
20			

8.2 Apêndice B - Questionário aplicado aos usuários

Neste apêndice encontra-se o questionário aplicado via web aos usuários.

Teste com usuários

Formulário sobre uso do aplicativo desenvolvido em projeto de TCC de Diego Souza de Mello Affonso. Todos os dados preenchidos aqui destinam-se única e exclusivamente à avaliação do aplicativo. Os resultados serão apresentados de forma a garantir o anonimato de cada resposta.

Perfil do Usuário

1. Nome:
2. Faixa etária:
    - Menos de 18 anos
    - 18 - 24 anos
    - 25 - 34 anos
    - 35 - 44 anos
    - 45 - 54 anos
    - Mais de 54 anos
3. Gênero:
    - Masculino
    - Feminino
    - Não-Binário
    - Prefiro não dizer
    - Outro
4. Frequenta academia?
    - Sim, 1-2x por semana
    - Sim, 3-5x por semana
    - Sim, 6-7x por semana
    - Não, mas pratico outras atividades físicas (corrida, crossfit, etc)
    - Não, e não pratico outras atividades físicas
5. Consulta com nutricionista?

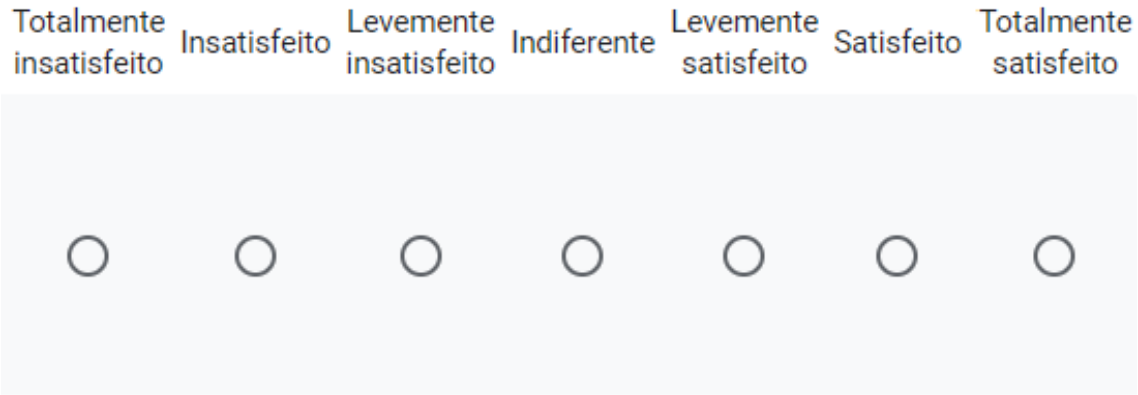
- Sim
- Não, mas já consultei
- Não, mas tenho interesse
- Não, e não tenho interesse

Tarefas

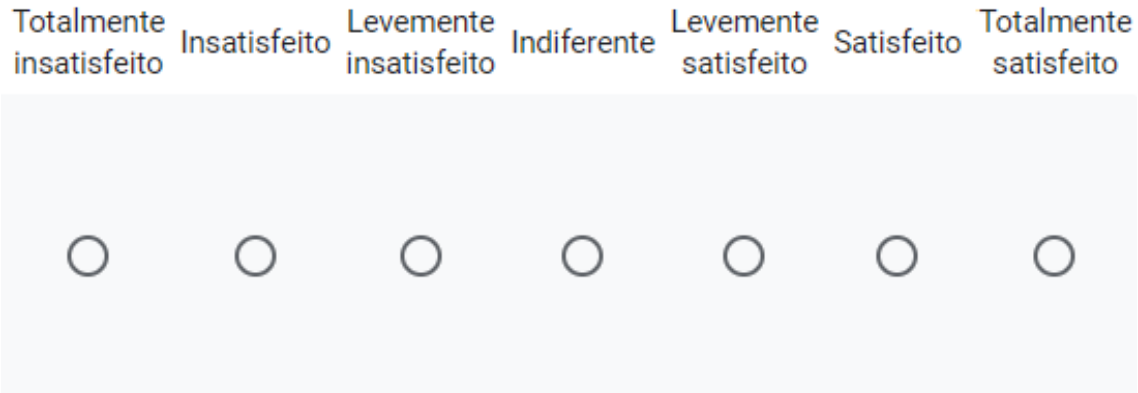
Tarefa 1: Abra o aplicativo entre como usuário. Em seguida, acesse “Avaliação pessoal” e registre uma avaliação boa (cara feliz). Retorne ao menu principal, acesse “Treinos” e confira o Treino B. Vá para “Plano Alimentar” e verifique o cardápio das 17h. Vá para “Metas” e confirme a meta ’Tomar 3L de água’. Retorne ao menu principal, acesse “Histórico de treinos”, selecione ’Tudo’ e confira o treino de 26/06/2022. Retorne ao menu principal e acesse “Perfil”.

- Quanto tempo você levou para a realização da tarefa?
- Classifique como se sente em relação às afirmações sobre a tarefa:

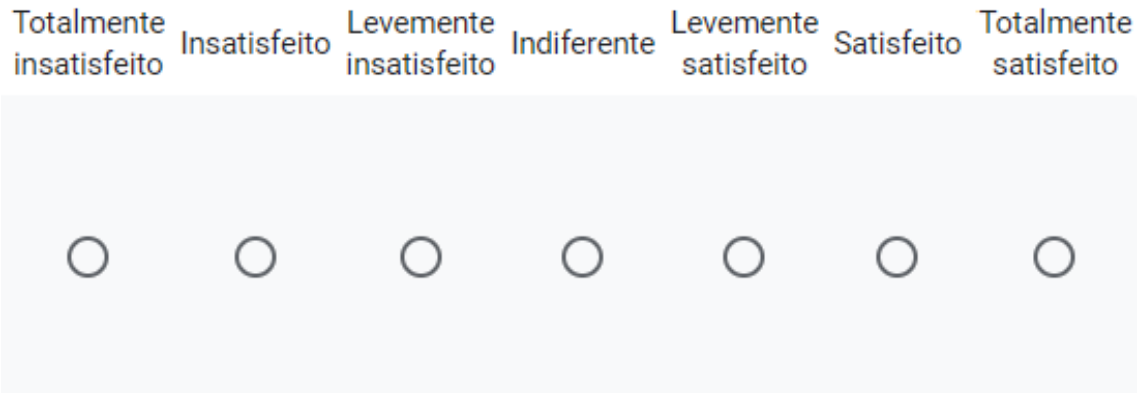
1. Estou satisfeito quanto a facilidade para completar a tarefa.



2. Estou satisfeito quanto ao tempo que levou para completar a tarefa.



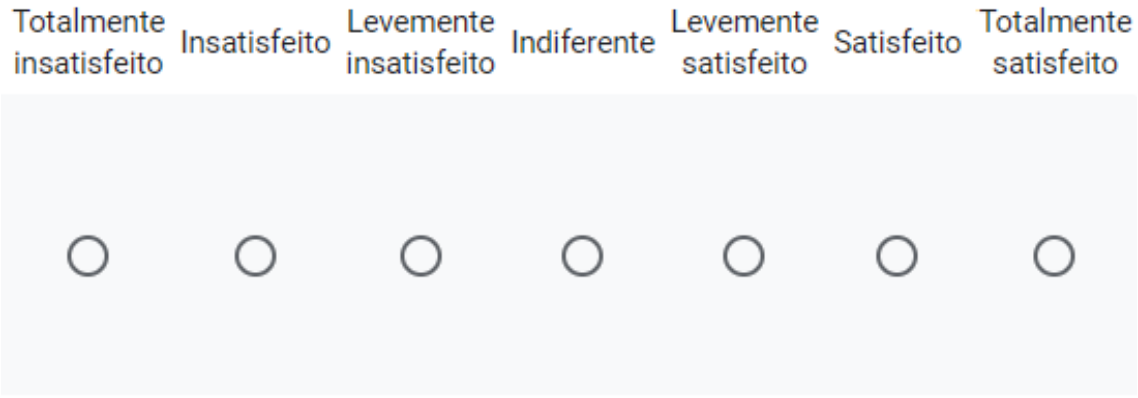
3. Estou satisfeito com o nível de suporte e informação enquanto completava a tarefa.



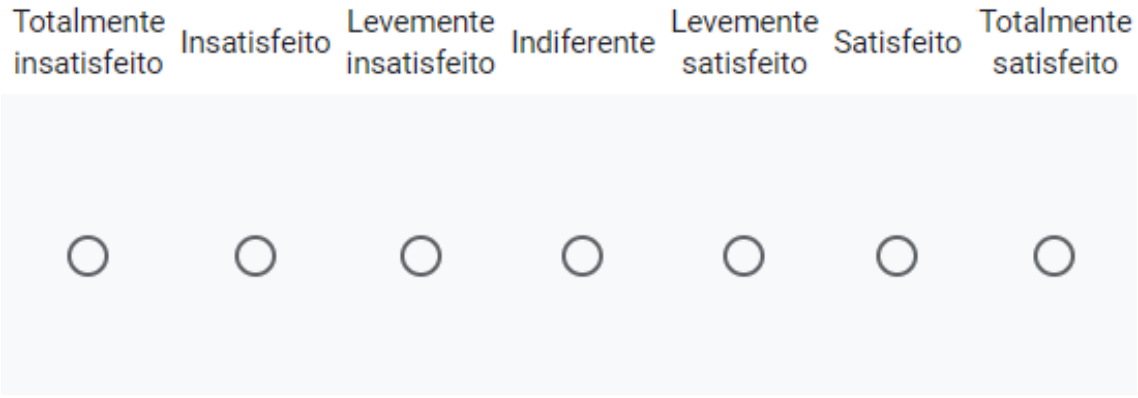
Tarefa 2: Abra o aplicativo e entre como profissional. Em seguida, selecione o paciente já criado Diego Souza e visualize sua alimentação das 12h. Retorne e visualize seu treino B. Retorne até a tela “Pacientes”, vá para “Criar consulta” e agende uma consulta para 12 de fevereiro de 2024, às 10:45, no nome de “Fu Lano”. Acesse a tela “Calendário” e confira se o agendamento foi realizado (parte inferior). Retorne ao menu principal e acesse a página “Perfil”.

- Quanto tempo você levou para a realização da tarefa?
- Classifique como se sente em relação às afirmações sobre a tarefa:

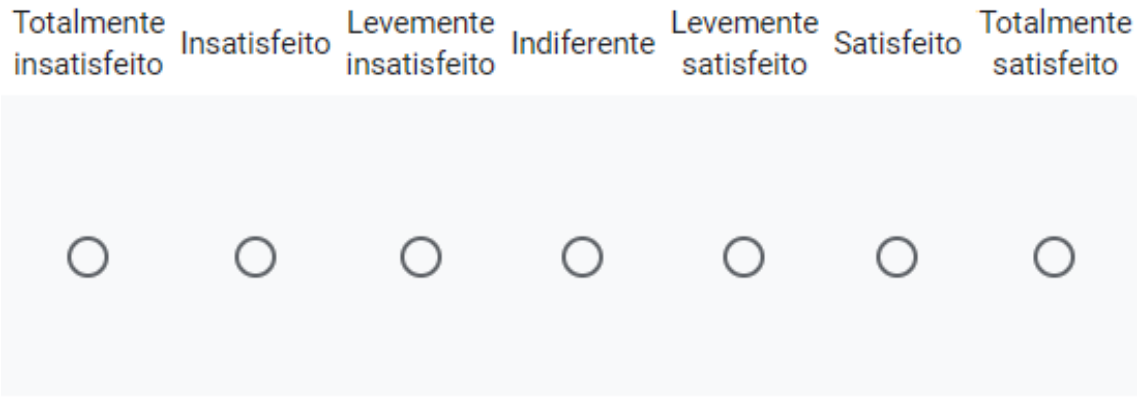
1. Estou satisfeito quanto a facilidade para completar a tarefa.



2. Estou satisfeito quanto ao tempo que levou para completar a tarefa.



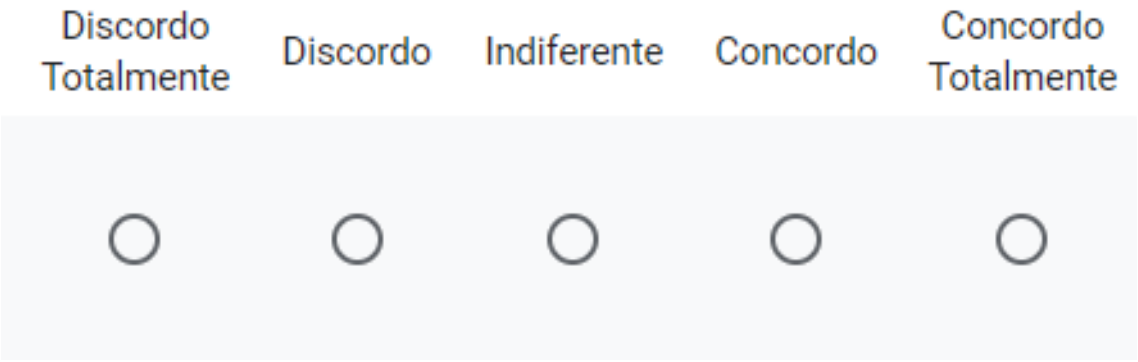
3. Estou satisfeito com o nível de suporte e informação enquanto completava a tarefa.



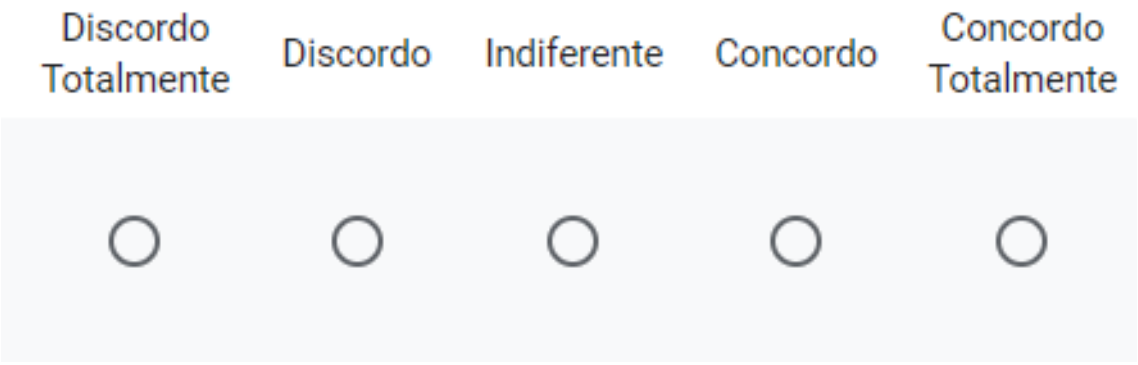
Avaliação do usuário

- Classifique como se sente em relação às afirmações sobre o aplicativo:

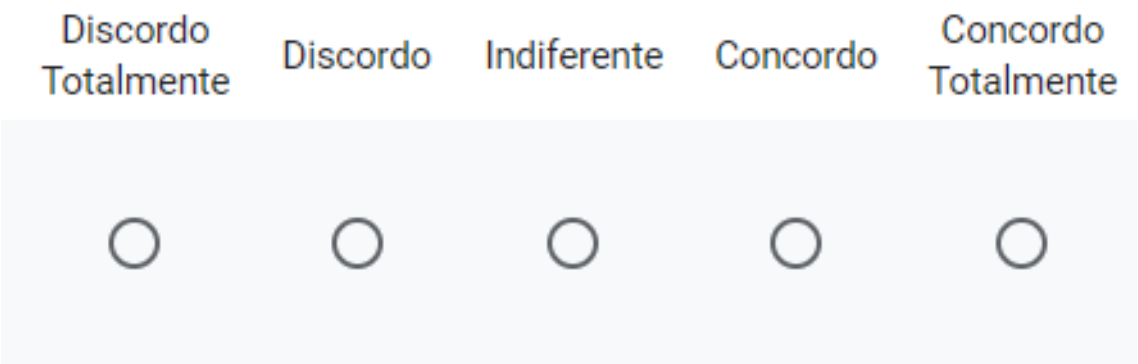
1. Eu acho que gostaria de usar esse sistema com frequência.



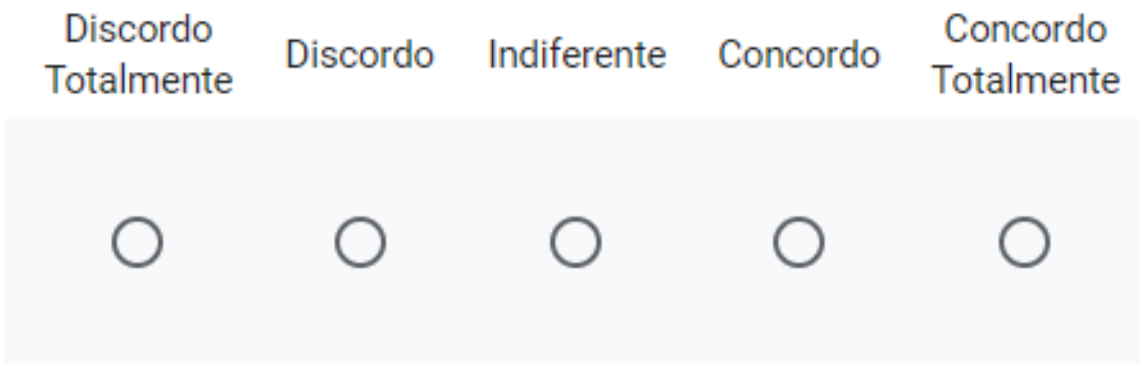
2. Eu acho o sistema desnecessariamente complexo.



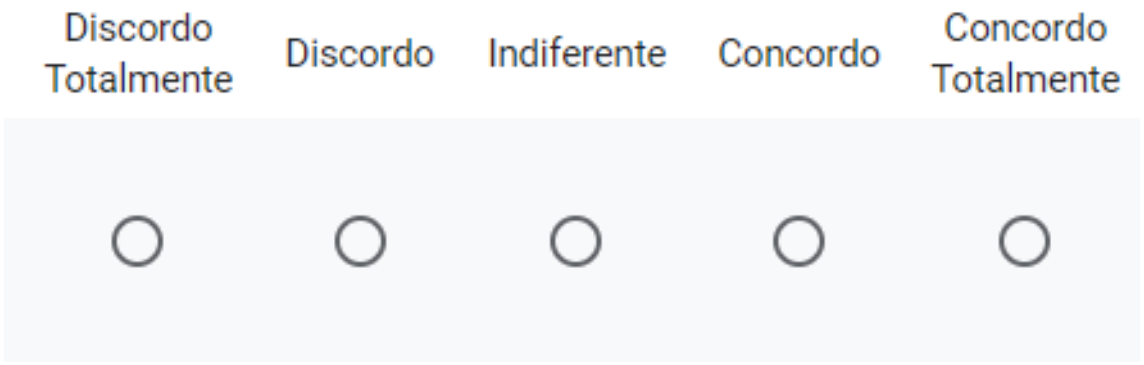
3. Eu achei o sistema fácil de usar.



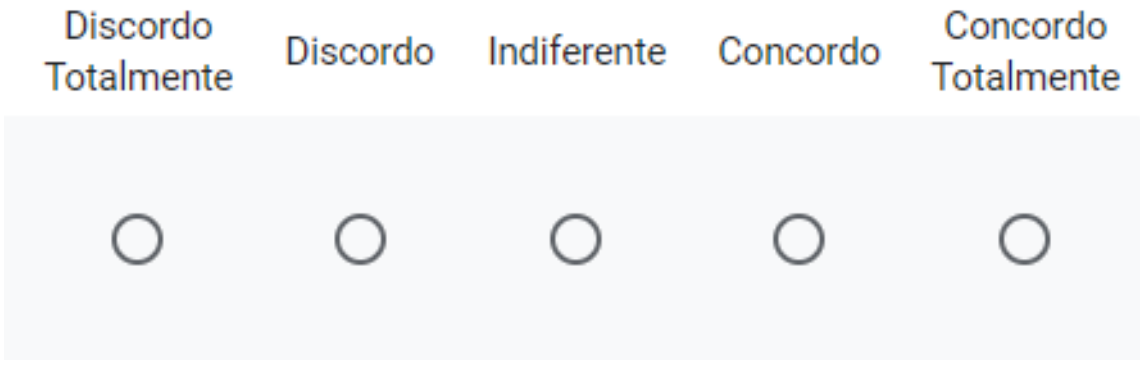
4. Eu acho que precisaria de ajuda de uma pessoa com conhecimentos técnicos para usar o sistema.



5. Eu acho que as várias funções do sistema estão muito bem integradas.



6. Eu acho que o sistema apresenta muita inconsistência.



7. Eu imagino que as pessoas aprenderão como usar esse sistema rapidamente.

Discordo Totalmente	Discordo	Indiferente	Concordo	Concordo Totalmente
○	○	○	○	○

8. Eu achei o sistema atrapalhado de usar.

Discordo Totalmente	Discordo	Indiferente	Concordo	Concordo Totalmente
○	○	○	○	○

9. Eu me senti confiante ao usar o sistema.

Discordo Totalmente	Discordo	Indiferente	Concordo	Concordo Totalmente
○	○	○	○	○

10. Eu precisei aprender várias coisas novas antes de conseguir usar o sistema.

Discordo Totalmente	Discordo	Indiferente	Concordo	Concordo Totalmente
○	○	○	○	○

- Em uma escala de 1 a 5, qual é o seu nível de satisfação geral com o sistema?

	1	2	3	4	5	
Muito insatisfeito	○	○	○	○	○	Muito satisfeito

- Feedback opcional:

REFERÊNCIAS

MCCLARAN, S. R. **The effectiveness of personal training on changing attitudes towards physical activity**. Journal of Sports Science & Medicine, v. 2, n. 1, p. 10–14, 2003.

ANDERSON SILVA MOURA, C. .; DOS SANTOS, Y. M.; SILVA NETO, J. G. da .; CARNEIRO CAVALCANTE, S. K. C.; GOMES DE SOUSA, E. F.; MARCOS HONÓRIO FILHO, S. .; PEREIRA ALVES, M. E. .; DE FREITAS PEREIRA, T. E. .; FREITAS PEREIRA, T. C. de; MENDES DE BRITO , A. N. . **Os perigos das dietas milagrosas sem acompanhamento do profissional nutricionista**. RECIMA21 - Revista Científica Multidisciplinar - ISSN 2675-6218, [S. l.], v. 3, n. 2, p. e321106, 2022.

RIES, E. **The Lean Startup**. New York: Crown Business, 2011.

Joe Shestak. **Flutter Usage Statistics 2023: Unveiling the Rise of Cross-Platform Development**. 2023. Disponível em: <https://www.linkedin.com/pulse/flutter-usage-statistics-2023-unveiling-rise-joe-shestak/>

Shan You. **What are no-code platforms and how can they speed digitalization?**. 2023. Disponível em: <https://www.weforum.org/agenda/2023/08/no-code-platforms-speed-digitalization/>

BROOKE, John et al. **SUS-A quick and dirty usability scale**. Usability evaluation in industry, v. 189, n. 194, p. 4-7, 1996.

Will T. **Measuring and Iterpreting System Usability Scale (SUS)**. 2021. Disponível em: <https://uiuxtrend.com/measuring-system-usability-scale-sus/>

Greg Satell. **Here's What Most People Get Wrong About Minimum Viable Products**. 2019. Disponível em: <https://greg-satell.medium.com/heres-what-most-people-get-wrong-about-minimum-viable-products-9de53ec6e6ea>

Nykolle Malone de Araujo. **React Native vs Flutter: qual escolher?**. 2023. Disponível em <https://community.revelo.com.br/react-native-vs-flutter-qual-escolher/>

Andryely Pedroso. **Top 5 softwares que todo nutricionista deveria conhecer**. 2021. Disponível em: <https://www.linkedin.com/pulse/top-5-softwares-que-todo-nutricionista-deveria-conhecer-pedroso/>

Jannik Lindner. **Must-Know Gym Injuries Statistics [Recent Analysis]**. 2023. Disponível em: <https://gitnux.org/gym-injuries-statistics/>

https://docs.flutter.dev

https://docs.thunkable.com