Universidade Federal de Ouro Preto
Instituto de Ciências Exatas e Aplicadas
Departamento de Computação e Sistemas

Evolução do Aplicativo para Montagem de Dietas UFITNESS

Gilberto Carlos Marçal

TRABALHO DE CONCLUSÃO DE CURSO

ORIENTAÇÃO:
Diego Zuquim Guimarães Garcia

Agosto, 2021
João Monlevade–MG

Gilberto Carlos Marçal

Evolução do Aplicativo para Montagem de Dietas UFITNESS

Orientador: Diego Zuquim Guimarães Garcia

Monografia apresentada ao curso de Sistemas de Informação do Instituto de Ciências Exatas e Aplicadas, da Universidade Federal de Ouro Preto, como requisito parcial para aprovação na Disciplina "Trabalho de Conclusão de Curso II".

Universidade Federal de Ouro Preto

João Monlevade

Agosto de 2021

SISBIN - SISTEMA DE BIBLIOTECAS E INFORMAÇÃO

M299e Marçal, Gilberto Carlos .
Evolução do aplicativo para montagem de dietas UFITNESS. [manuscrito] / Gilberto Carlos Marçal. - 2021.
53 f.: il.: color., tab..

Orientador: Prof. Dr. Diego Zuquim Guimarães Garcia.
Monografia (Bacharelado). Universidade Federal de Ouro Preto. Instituto de Ciências Exatas e Aplicadas. Graduação em Sistemas de Informação .

1. Aplicativos móveis. 2. Dieta. 3. Hábitos de saúde - Software educacional. 4. Software de aplicação - Desenvolvimento. I. Garcia, Diego Zuquim Guimarães. II. Universidade Federal de Ouro Preto. III. Título.

CDU 004.41

Bibliotecário(a) Responsável: Flavia Reis - CRB6-2431

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E APLICADAS
DEPARTAMENTO DE COMPUTACAO E SISTEMAS

FOLHA DE APROVAÇÃO

Gilberto Carlos Marçal

Evolução do aplicativo para montagem de dietas UFITNESS

Monografia apresentada ao Curso de Sistemas de Informação da Universidade Federal
de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação

Aprovada em 31 de agosto de 2021

Membros da banca

Doutor - Diego Zuquim Guimarães Garcia - Orientador - Universidade Federal de Ouro Preto
Mestra - Daniela Rodrigues Dias - Universidade Federal de Ouro Preto
Mestre - Euler Horta Marinho - Universidade Federal de Ouro Preto

Diego Zuquim Guimarães Garcia, orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 09/11/2021

Documento assinado eletronicamente por **Diego Zuquim Guimaraes Garcia**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 09/11/2021, às 22:34, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0242965** e o código CRC **C6CE861A**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35400-000
Telefone: - www.ufop.br

Este trabalho é dedicado aos meus familiares, amigos e companheiros que estiveram ao meu lado na árdua jornada universitária e aos meus orientadores durante todo o curso pelo incentivo e apoio constantes.

Agradecimentos

Agradeço a todos aqueles que, de alguma forma, contribuíram para a realização deste trabalho.

Aos meus pais e irmãos, que me incentivaram nos momentos difíceis e compreenderam a minha ausência enquanto eu me dedicava à realização deste trabalho.

Aos amigos e companheiros que convivi ao longo desses anos de curso, me incentivando e que certamente tiveram impacto na minha formação acadêmica.

"Não espere por uma crise para descobrir o que é importante em sua vida."

— Platão (427-347 a.C.)

Resumo

A tecnologia vem sendo aprimorada e se popularizado em todos os âmbitos da nossa das nossas vidas, está simplificando os processos. Hoje tem se popularizado a criação de aplicativos para tudo, exemplo disso temos os Apps de transporte particular que permitem em poucos cliques que o usuário encontre um motorista baseado em sua localização e os leve para qualquer outro lugar. É simples, rápido e eficaz. Outro ótimo exemplo que pode ser citado é a forma com a qual pedimos comida, que nos últimos tempos tem ganhado grande força os Apps Deliverys, principalmente nas grandes capitais palas vantagens que esse tipo de transporte nos traz. Seja com inovações na comida, nos modelos de entrega ou até mesmo na forma de fazer e finalizar sua compra. A tendência é que a tecnologia faça parte das nossas vidas cada vez mais. Na área da saúde não foi diferente, tivemos inúmeros avanços significativos. Com o aumento da disponibilidade da informação e facilidades para conseguir alimentos rápidos e cada vez mais baratos, se popularizou a alimentação por fast food que acaba tendo impacto negativo sobre a saúde das pessoas geralmente. De acordo com um estudo realizado entre 2014 a 2019 pela EAE Bussiness School o Brasil entrou para a lista dos países no mundo que mais consomem fast food, o crescimento chegou a incríveis 0,8% nesse período e esse valor pode ter sofrido grande aumento durante a pandemia e isolamento social. Com isso várias empresas estão desenvolvendo formas de conscientizar as pessoas sobre os possíveis malefícios que excesso podem trazer a saúde. Aplicativos que auxiliam na manutenção da saúde estão cada vez mais populares nas lojas como PlayStore e AppleSptre e já contam com milhares de downloads em sua totalidade.

Palavras-chaves: dieta. fitness. educação alimentícia. evolução de software. software multiplataforma.

Abstract

Technology has been improved and popularized in all areas of our lives, it is simplifying processes. Today the creation of apps for everything has become popular, an example of which we have the private transport Apps that allow the user to find a driver based on their location in a few clicks and take them to any other place. It's simple, fast and effective. Another great example that can be mentioned is the way in which we order food, which in recent times Apps Deliverys have gained great strength, especially in large capitals because of the advantages that this type of transport brings us. Whether with innovations in food, delivery models or even in the way you make and complete your purchase. The trend is that technology is part of our lives more and more. In the area of health it was no different, we had numerous significant advances. With the increasing availability of information and facilities to get fast and cheaper food, fast food eating became popular, which ends up having a negative impact on people's health in general. According to a study carried out between 2014 and 2019 by the EAE Bussiness School, Brazil entered the list of countries in the world that consume the most fast food, the growth reached incredible 0.8% in this period and this figure may have suffered large increase in the pandemic period and social isolation. With this, several companies are developing ways to make people aware of the possible harm that excess can bring to health. Apps that help maintain health are increasingly popular in stores like PlayStore and AppleSptre and already have thousands of downloads in their entirety.

Key-words: diet. fitness. food education. software evolution. cross-platform software

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

POF	Pesquisa de Orçamento Familiar
IBGE	Instituto Brasileiro de Geografia e Estatística
TACO	Tabela Brasileira de Composição de Alimentos
ENDEF	Estudo Nacional de Despesa Familiar
SDK	Software Development Kit
API	Application Programming Interface
PWA	Web Progressive Aplication
DOM	Document Object Model

Sumário

1 Introdução

Realizando breve analise em relação à alimentação da população nota-se que a dieta inadequada e inatividade física são os principais fatores associados às doenças crônicas não transmissíveis como obesidade, diabete do tipo 2, doenças coronárias e outras (FAO; WHO, 2003). Em contrapartida, temos os casos de deficiências nutricionais que também tem na dieta seu fator etiológico, como exemplo temos anemia. Toda dieta é passível de mudança, torna-se necessário o desenvolvimento de políticas para a prevenção, tanto das deficiências nutricionais, quanto das doenças crônicas não transmissíveis. Mudanças que podem ter por base o estudo e monitoramento de indicadores de consumo alimentar. Serra et al. (2003), em suas pesquisas informam que POF[1] contribuem como fontes valiosas de informação em relação aos indicadores de consumo familiar no Brasil.

Tem ocorrido no Brasil regulamente as POF's em áreas metropolitanas e por meio dessas tem sido possível avaliar melhor as tendências celulares de disponibilidade domiciliar de alimentos no país (MONDINI; MONTEIRO, 1994; MONTEIRO; MONDINI; LEVY-COSTA, 2000). Com as POF não é possível avaliar o consumo individual porem conseguimos realizar estimativas que são fundamentais para coleta de dados de dietas, apesar de que o foco principal seja estimar índices de preços. O objetivo de usar essa abordagem é que a metodologia é padronizada na coleta de dados, utiliza amostragem probabilística, são periódicas e incluem detalhada mensuração de características socioeconômicas. (LAGIOU; TRICHOPOULOU, 2001)

Para exemplificar como é feito o POF utilizaremos a realizada entre julho de 2002 a junho de 2003 e envolveu entrevistas realizadas numa amostra de 48470 domicílios. O plano de amostragem da pesquisa, bem próximo ao método utilizado pelo Instituto Brasileiro de Geografia e Estatística (IBGE) em pesquisas de orçamento familiar (IBGE, 2004). Em resumo, trata-se de amostragem por conglomerado em dois estágios com estratificação geográfica e socioeconômica das unidades primárias de amostragem correspondentes aos setores censitários da base geográfica do Censo Demográfico 2000. As unidades secundárias de amostragem foram os domicílios particulares permanentes no setor. Os setores censitários foram selecionados por amostragem sistemática com probabilidade proporcional ao número de domicílios no setor, enquanto as residências foram eleitas por amostragem aleatória simples, sem reposição, dentro dos setores censitários sorteados. Os setores sorteados e respectivos domicílios selecionados foram distribuídos ao longo de 12 meses de duração da pesquisa, garantindo-se em todos os trimestres a coleta de dados em todos os estratos geográficos e socioeconômicos. A amostragem da POF 2002 – 2003 foi estruturada para produzir estimativas representativas do País como um todo e de todas as unidades da

[1] POF. Pesquisa de Orçamento Familiar

Federação.

Durante sete dias consecutivos, a informação básica da POF[2] analisada compreende as aquisições alimentares e de bebidas para consumo domiciliar feitas pela unidade de consumo (família) e registradas diariamente pelo morador do domicílio ou pelo entrevistador do IBGE[3]. No registro inclui-se a descrição detalhada do produto, a quantidade adquirida e a unidade de medida, além do valor da despesa em Reais, local de compra e forma de aquisição. Na maioria dos casos foi possível apurar precisamente a quantidade em KG ou Litros adquiridos e nos caso onde não foi possível realizar essa apuração leva-se como base o valor da despesa e do preço médio do produto.

Os indicadores empregados incluem a média do valor calórico total da disponibilidade alimentar domiciliar (expressa em kcal per capita por dia) e a participação relativa, na disponibilidade alimentar de alimentos, grupos de alimentos e nutrientes selecionados.

Para realizar a apuração calórica foram aplicadas algumas técnicas que ajudam a calcular o valor mais aproximado para cada nutriente. Para se chegar às quantidades disponíveis de calorias e macro nutrientes, foram utilizadas três tabelas de composição alimentar: a tabela TACO[4] (Tabela Brasileira de Composição de Alimentos)(Universidade Estadual de Campinas [NEPA/ Unicamp], 2004); a tabela Guilherme Franco (FRANCO, 1992) e a tabela IBGE/Estudo Nacional de Despesa Familiar (ENDEF). A tabela TACO se destacou na utilização do cálculo dos alimentos de origem animal, para embutidos a tabela Guilherme Franco e os demais alimentos a tabela do IBGE/ENDEF.

Como resultado dessa POF chegou-se na disponibilidade média domiciliar de alimentos no Brasil estimada em 1.800 kcal por pessoa por dia, sendo essa disponibilidade próxima de 1.700 kcal no meio urbano e de 2.400 kcal no meio rural (Tabela 1). Ressalva-se que não é possível avaliar a adequação dessa disponibilidade calórica, uma vez que não se dispõe de uma avaliação direta dos alimentos efetivamente consumidos pelas famílias, bem como das quantidades de alimentos consumidos fora do domicílio.

Na Tabela 1 foi mostrado a participação dos alimentos e dos grupos alimentares tendo os alimentos básicos de origem vegetal correspondendo a cerca de 50% das calorias totais, alimentos essencialmente calóricos com 28%, produtos de origem animal 18%, as frutas e verduras correspondem a 2,3% das calorias.

A participação de cereais de derivados nas áreas urbanas e rurais foi bem próxima onde se destacou a participação do pão, biscoito e macarrão em áreas urbanas e arroz e farinha de trigo nas áreas rurais. Houve maior participação na dieta de frutas, verduras e legumes, carnes, leite e derivados no meio urbano, enquanto no meio rural houve maior participação de feijões e outras leguminosas e de raízes e tubérculos. A participação na

[2] POF. Pesquisa de Orçamento Familiar
[3] IBGE. Instituto Brasileiro de Geografia e Estatística
[4] TACO. Tabela Brasileira de Composição de Alimentos

dieta de refeições prontas e misturas industrializadas foi três vezes maior no meio urbano do que no meio rural (Tabela 1).

Graças a realização da POF 2002 – 2003 em larga escala em todo território Brasileiro, foi possível em primeira mão analisar e mapear a situação da distribuição regional e socioeconômica de importantes indicadores do padrão alimentar.

Esse e vários outros estudos motivaram diversos avanços nas questões alimentares e motivou a criação de tecnologias que auxiliassem na compreensão dos alimentos e do impacto que eles têm sobre nossas vidas. Esse presente trabalho busca explorar uma alternativa tecnológica para montar dietas com o máximo de informações calóricas para usuários de smartphones Androide. O presente trabalho trata de uma evolução do *software* já existente UFitness[5] lançado no mercado em 2020.

[5] Disponível em <https://play.google.com/store/apps/details?id=com.ufitness.too&hl=pt_BR>

Tabela 1 – Participação Relativa (%) de alimentos e grupos de alimentos no total de calorias determinado pela aquisição alimentar domiciliar por situação do domicílio. Brasil, 2002/2003

Grupos de alimentos	Total	Situação do domicílio	
		Urbano	Rural
Cereais e derivados	36,4	36,4	36,3
Arroz polido	17,8	17,3	19,3
Pão francês	5,5	6,5	1,9
Biscoitos	3,1	3,3	2,4
Macarrão	2,7	2,9	2,0
Farinha de trigo	2,8	2,5	3,8
Outros	4,6	3,9	6,8
Feijões e outras leguminosas	6,6	5,8	9,0
Raízes, tubérculos e derivados	5,8	4,3	10,6
Batata	0,7	0,8	0,6
Mandioca	0,4	0,2	0,9
Outros	4,6	3,3	9,2
Carnes	11,8	12,3	10,0
Bovina	5,1	5,4	4,1
Frango	2,5	2,7	1,8
Suína	1,2	1,1	1,7
Peixes	0,6	0,5	1,0
Embutidos	2,2	2,5	1,1
Outras	0,1	0,1	0,3
Leites e derivados	6,3	6,7	5,1
Leite	4,6	4,6	4,4
Queijos	1,1	1,3	0,5
Outros	0,6	0,7	0,1
Ovos	0,3	0,3	0,5
Frutas e sucos naturais	1,6	1,8	0,9
Bananas	0,7	0,7	0,4
Laranjas	0,2	0,2	0,1
Outras	0,8	0,9	0,3
Verduras e legumes	0,7	0,8	0,5
Tomate	0,2	0,2	0,1
Outros	0,6	0,6	0,4
Óleos e gorduras vegetais	12,8	13,5	10,4
Óleo de soja	10,5	10,9	9,3
Margarina	1,8	2,1	0,8
Outros	0,5	0,6	0,3
Gordura animal	1,3	1,2	1,8
Manteiga	0,4	0,4	0,2
Toucinho	1,0	0,8	1,7
Açúcar e refrigerantes	13,4	13,4	13,3
Açúcar	11,9	11,7	12,8
Refrigerantes	1,5	1,7	0,6
Bebidas alcoólicas	0,5	0,5	0,2
Cerveja	0,3	0,4	0,1
Aguardente	0,1	0,1	0,1
Outras	0,1	0,1	0,0
Oleaginosas	0,2	0,1	0,4
Condimentos	0,6	0,7	0,3
Refeições prontas e misturas industrializadas	1,7	2,0	0,9
Total	100,0	100,0	100,0
Total de calorias (kcal/dia *per capita*)	1.811	1.690	2.402

Fonte – Rev Saúde Publica 2005;39(4):530-40

1.1 Definição do problema

No atual cenário, onde as pessoas têm puco tempo para as tarefas cotidianas, comidas como fast food que proporcionam maior rapidez no preparo, são uma realidade. Com isso a qualidade da alimentação benéfica para o corpo acaba sendo deixada de lado. Fato este mostrado pelos estudos recentes realizados pelo Ministério da Saúde já citado no início deste texto. Outro fator que impede que uma pessoa busque por hábitos alimentares mais saudáveis está ligado a falta de disciplina, onde a mesma abandona as recomendações e acompanhamento passado por nutricionistas, por exemplo, já que ele não está ao seu lado o tempo todo. Com base nisso, a aplicação desenvolvida neste trabalho tem como foco proporcionar às pessoas um guia para que os usuários consigam alcançar seu peso ideal e se alimentem de forma mais consciente.

Nesse sentido é importante frisar que a aplicação desenvolvida está em processo de mudanças e fatores tecnológicos possibilitaram que novas funcionalidades fossem incrementadas ao Software UFitness dando origem a sua segunda versão.

1.2 Problema de pesquisa

De acordo com Svahnberg (2003), o sucesso de um software em uma perspectiva a longo prazo não é determinado em sua versão inicial, o sucesso é definido pala capacidade de adaptabilidade e evolução no atendimento das novas exigências do mercado considerando os requisitos funcionais e os de qualidade. Já para o autor (BASS et al., 1998; BOSCH, 2000), o sucesso a logo prazo tem mais afinidade com o atendimento dos requisitos de qualidade que são atualizados continuamente ao passar do tempo.

No desenvolvimento inicial do software, os requisitos de qualidade extensibilidade e adaptabilidade são muito importantes para aumentar a quantidade de recursos suportados pelo software. Após algum tempo começa a surgir novas requisições de qualidade como confiabilidade e desempenho que devem ser incorporados na evolução do projeto. Essa etapa sugere que os atributos sejam implementados para melhorar a sua visibilidade de mercado uma vez que outras opções concorrentes estão amplamente disponíveis para os usuários, melhorando assim sua competitividade. Caso um produto esteja estagnado e não evolua com as tendências impostas pelo mercado, o mesmo está sujeito a ser substituído por outro que atenda às requisições, tendo como resultado seu fim não programado. (SVAHNBERG, 2003)

Kruchten, Obbink e Stanford (2006) afirmam que existe a possibilidade de supervisionar e controlar o desenvolvimento e evolução de sistemas através de arquitetura de software.

De acordo com Portes (1997), todo software está em contínua evolução a partir de

mudanças consecutivas com objetivo central de correção de erros, aumento de desempenho ou mesmo implementação de novos requisitos adquiridos pelo tempo, ou pelo mercado.

Um software pode ter seu fim prematuro ou cair em desuso caso o mesmo pare de evoluir, ou pare de atender as mudanças de origem sociais, ou ambientais. As mudanças sociais estão relacionadas a mudanças funcionais e as ambientais, normalmente associadas a arquitetura do software. De modo geral, a evolução deve ser um processo de aprimoramento constante, ágil e que não degrade o objetivo inicial do projeto.

Para Sommerville (2011), a evolução dos processos de um software podem variar de acordo com a organização e pode ser obtida de diversas formas desde as mais complexas e documentadas até as mais informais como conversas de usuários do sistema e desenvolvedores.

Quando se trata dos termos evolução, manutenção e erosão existe uma sutil diferença semântica entre elas. Essa diferença é apontada por Parnas (1994) em que manutenção implica em manter um software rodando sem mudar seu aspecto físico, ou seja, alterando apenas partes desgastadas em sistemas físicos. Na prática, a mudança no designe é sim uma forma válida de manutenção quando se trada de software até mesmo para correção de falhas visuais e para se adequar a novos ambientes. A erosão é a consequência obtida pela manutenção realizada de forma incorreta.

Já a evolução sugere uma mudança que não está bem definida na manutenção descrita anteriormente e em muitos casos, associada a várias mudanças sendo visuais e funcionais, dessa forma podemos diferenciar em termos mais simples a evolução do software sendo um conglomerado de mudanças que o software sofre para se manter vivo no mercado indo muito além de correção de erros estéticos e funcionais. (GODFREY; GERMAN, 2008)

O autor TU e GODFREY (2002) acredita que a existem vários desafios quando se trata da evolução do software e para se ter êxito, a arquitetura deve ser identificada e gerenciada para se manter a coerência estrutural. Dessa forma o impacto negativo gerado pela manutenção, onde há degradação da vida e da confiabilidade, pode ser revertido em tempo de vida ao software com o adequado planejamento.

Algumas consequências na confusão dos termos manutenção e evolução do software é a pouca atenção dada aos modelos de processos que dão suporte a evolução continuada de sistema de longo tempo de vida. A exemplo temos o modelo em cascata, que lida com a manutenção apenas como um estágio final no desenvolvimento. Temos também o modelo iterativo incremental que sugere que a evolução ocorra durante as consecutivas interações no desenvolvimento.

1.3 Objetivo Geral e Objetivos específicos

O objetivo desse trabalho é desenvolver uma atualização do Aplicativo UFITNESS implementando novas funcionalidades e novas disposições. O aplicativo mantém a proposta de montar dietas e fornecer informações úteis para que os usuários entendam os conceitos básicos referentes a perda de peso.

O Aplicativo terá várias funcionalidades, desde a criação dos planos de dietas em si, cálculo de I.M.C. e Taxa metabólica basal, consulta tabela TACO e mudanças no layout. Os usuários não precisarão efetuar cadastro no aplicativo para utilizarem suas funcionalidades.

Abaixo serão descritos alguns critérios que serão seguidos para o desenvolvimento da aplicação e assim alcançar o objetivo do aplicativo:

- A aplicação será desenvolvida para a plataforma mobile (tablets e smartphones), uma vez que o número de usuários de aplicações em dispositivos vem apresentando crescimento cada vez maior.;
- O sistema contará com seções de ajuda para ajudar os usuários a entenderem conceitos importantes para a construção dos planos alimentares.;
- Consulta de alimentos da tabela TACO (Tabela Brasileira de Composição de Alimentos);

1.4 Resultados Esperados

Esse trabalho teve como objetivo principal implementar o recurso de consulta de alimentos via meios externos utilizando-se as tecnologias de desenvolvimentos mobile multiplataforma considerando a arquitetura utilizada no desenvolvimento e estudando possíveis melhorias visando os requisitos de qualidade.

1.5 Estrutura da Monografia

Este trabalho está estruturado conforme descrito a seguir. O Capítulo 2 trata da fundamentação teórica do trabalho, onde os principais conceitos e definições são realizados. Nesse mesmo capitulo é abordado as tecnologias que foram usadas para o desenvolvimento, suas principais características e suas respectivas aplicações. O Capítulo 3 tem como objetivo realizar uma análise das possíveis mudanças, aplicação da metodologia ATAM (adaptada) e também demonstrar como a API principal de busca funciona. São apresentados no Capítulo 4 os resultados obtidos com a implementação das mudanças. Por último, no Capítulo 5 são encontradas as conclusões.

2 Revisão Bibliográfica

Atualmente temos uma vasta diversidade de dispositivos tecnológicos que utilizam sistemas operacionais ou arquiteturas distintas, porém, contendo os mesmos aplicativos e compartilham diversas funcionalidades. Há certo tempo cada sistema/arquitetura tinha, para um mesmo software, linhas de desenvolvimento parcialmente independentes, pois, para cada arquitetura é desenvolvido softwares compatíveis respeitando as particularidades de cada uma. Esse processo independente de desenvolvimento pode ser muito custoso para os desenvolvedores e/ou demorado, já que conta com o desenvolvimento em paralelo e muitas vezes com equipes diferentes que tomam decisões perante as limitações ou requisitos das arquiteturas diferentes trabalhadas.

Mudanças são eventos inevitáveis em quase todos os tipos de softwares existentes. Isso ocorre devido a mudanças dos requisitos no meio externo que precisam ser sanadas ao longo do tempo. Com o passar do tempo, novas tecnologias vão surgindo e por esse motivo os softwares devem estar prontos para acomodar eventuais mudanças tecnológicas e humanas. Essas mudanças que são realizadas nos softwares geram custos, pois, exige a necessidade de retrabalho.(SOMMERVILLE, 2011)

Uma das principais estratégias adotadas nesse cenário é contratação de uma segunda equipe de desenvolvimento para otimizar os processos e os recursos no desenvolvimento das aplicações. Adotando essa estratégia normalmente o resultado é uma disponibilização mais rápida nas plataformas pretendidas, um gerenciamento de versão mais apurado, uma equipe de suporte mais alinhada na resolução de problemas e troca de informações para futuras mudanças. Outro resultado direto é o aumento do custo do desenvolvimento, aliás, agora existem duas equipes ou mais para desenvolver uma mesma aplicação entrando no faturamento da empresa como despesa com pessoal. A alternativa para esse impasse seria o que algumas empresas de desenvolvimento aderem: desenvolvimento sequencial. Nessa modalidade a empresa mantêm seu quadro de desenvolvedores iniciais porem como consequência direta há o aumento significativo no prazo de entrega das demais plataformas de desenvolvimento. Ambas alternativas citadas são ditas como ineficientes nos dias atuais, pois, a cada dia que passa a exigência do mercado é sempre entregar os trabalhos com prazos cada vez menores e os custos de desenvolvimentos também devem ser reduzidos. (EL-KASSAS; WAFAA, 2017)

Diante da necessidade latente de desenvolver aplicações multiplataformas que o mercado impôs, foram surgindo frameworks de desenvolvimento multiplataforma que possibilitam a implementação de uma única base de código que possa ser interpretada em plataformas diferentes da mesma forma.

2.1 Desenvolvimento Paralelo

Para El-Kassas e Wafaa (2017), o desenvolvimento paralelo permite que uma aplicação seja desenvolvida e possa ser utilizada em qualquer outra plataforma sem a necessidade de adaptações bruscas. Já existem várias iniciativas de frameworks com essas propostas porem, a maioria ainda não teve seu uso comercial liberado por estarem em desenvolvimento e pesquisas ou os que já estão liberados, mas não conseguem contemplar aplicações mais robustas no desenvolvimento multiplataforma.

Várias soluções estão disponíveis no mercado para se entregar aplicações em plataformas distintas, cada abordagem possui suas particularidades. A escolha da ideal deve ser feita avaliando os objetivos do negócio, necessidades que possam apresentar na aplicação desejada e as particularidades da abordagem. Independente de qual abordagem for adotada será necessária uma análise do negócio anteder a modelagem para o desenvolvimento. Essa análise tem como objetivo de abstrair ao máximo os requisitos dos stakeholders independentes das dificuldades que possam representar nas etapas de design e desenvolvimento.

Biørn-Hansen, Grønli e Ghinea (2018) em suas obras apontam cinco categorias principais que abordam o desenvolvimento multiplataforma e variados frameworks, outros menos citados na comunidade de desenvolvedores foram ignorados. As abordagens discutidas foram: Híbrida, Interpretada, compilação multiplataforma, orientada a modelos e aplicações web progressivas.

2.1.1 Abordagem: Desenvolvimento Híbrido

Essa abordagem possibilita o uso de tecnologias web como HTM, CSS e JavaScript para o desenvolvimento das aplicações destinadas a mais de uma plataforma. Quando o cenário é mobile, funciona da seguinte forma: existe uma aplicação nativa que inicializa as aplicações do código principal como componentes WebView, essa renderização ocorre de forma similar a um navegador (web) porem os arquivos criados são interpretados com os componentes nativos de cada plataforma especifica.

O código criado contém o fundamento lógico para que a aplicação consiga executar suas funções, bem com o constructo do layout. O resultado geral da utilização dessa abordagem é um pacote de aplicação (package) de arquivos com definições da lógica, interfaces e o fundamental que é o código responsável por fazer interpretação da webview na plataforma nativa. Pelo fato da aplicação web estar sendo executada a partir da própria arquitetura nativa, o desenvolvedor não tem problemas ao disponibilizar o mesmo nas principais lojas como App Store e Play Store e esse processo seria o mesmo ou quase o mesmo utilizado ao disponibilizar aplicativos pelos ambientes de desenvolvimento oficial.

Várias vantagens podem surgir da utilização do desenvolvimento hibrido, pois, aqui é

utilizado tecnologias no padrão web que já são bem conhecidas e tem vasta compatibilidade, o código pode ser facilmente reaproveitado para o caso de querer disponibilizar a mesma aplicação no ambiente web. Outra grande vantagem é que o desenvolvedor com grande experiência no desenvolvimento Web pode, sem muito esforço, iniciar a expansão de seus trabalhos para o mercado mobile que está cada dia mais sendo visado. Uma das desvantagens apresentadas nesse método é que por ser uma aplicação emulada Web os componentes nativos do layout nem sempre podem ser utilizado podendo impactar assim na usabilidade dos usuários finais. Para resolução da maioria desses problemas foram desenvolvidos vários Frameworks que tem a função de interpretar o código em componentes nativos, normalmente trabalham com tecnologias CSS, HTML e JavaScript.

2.1.2 Abordagem: Desenvolvimento Interpretado

Essa abordagem é semelhante à abordagem Hibrida e é muito comum encontrar frameworks que permitem o desenvolvimento de aplicações utilizando a linguagem JavaScript. Aprincipal diferença entre essas abordagens é que a primeira faz uso de componentes WebView para renderizar a aplicação desenvolvida já a segunda, faz uso de interpretadores Javascript presentes nos dispositivos para ter acesso aos componentes nativos de cada plataforma bem como outras integradas como câmera, GPS e outros. A comunicação entre o framework e a arquitetura nativa se dá por meio der uma técnica conhecida como bridging.

A principal vantagem de criar aplicações utilizando essa abordagem é que os componentes nativos de cada plataforma são renderizados dispensando assim numerosas linhas de código que podem simular algo parecido, o usuário final não consegue distinguir essa aplicação de outras criadas em ambientes oficiais. Um dos únicos problemas observados é que como o código original tem que ser lido e interpretado em código de outra linguagem ou arquitetura, existe uma perda de performance durante a execução.

2.1.3 Abordagem: Compilação multiplataforma

Diferentemente das abordagens Hibrida e Interpretada, os frameworks que foram construídos para trabalhar com a abordagem multiplataforma não depende de recursos de interpretação ou Webview que rodam em tempo de execução. Aqui o objetivo é utilizar uma única linguagem de programação que tem acesso a recursos das API's nativas dos dispositivos alvos através de SDK's (Kits de Desenvolvimento de Softwares) contidos nos frameworks, que são responsáveis por mapear as funcionalidades desenvolvidas para as funcionalidades específicas em cada plataforma. O resultado final desse modelo é a geração de um arquivo de Bytecode contendo informações da linguagem original que por fim será compilado e irá gerar um executável para cada uma das plataformas alvo.

A grande vantagem na utilização dessa abordagem é a capacidade de realizar o desenvolvimento multiplataforma utilizando-se uma única linguagem de programação de forma geral e conseguir entregar aplicações nativas mantendo aparência e desempenho semelhantes ou muito próximo do que é possível fazer com os ambientes de desenvolvimentos padrões. A desvantagem desse modelo é que só é possível desenvolver aplicações que são previamente compatíveis com o framework, ou seja, caso surja uma nova plataforma que se pretende desenvolver é necessário aguardar até o framework atualizar e forneça compatibilidade para o novo ambiente.

2.1.4 Abordagem: Orientada a modelos

A origem dessa abordagem é o paradigma de desenvolvimento orientado a modelos (Model-Driven Development) e consiste na criação de aplicação utilizando linguagens especificas disponibilizadas pelos frameworks, as chamadas linguagens específicas de domínio. O objetivo é abstrair o problema para ser resolvido de forma textual ou gráfica de modo a facilitar a visualização das informações ao ponto onde desenvolvedores e não desenvolvedores consigam interpretar.

A grande vantagem desse modelo é que com ele é possível que desenvolvedores e pessoas comuns possam construir aplicações a partir do domínio e não a partir da linguagem de programação, dessa forma é possível que uma pessoa consiga compilar regras de negócio em aplicação que pode ser disponibilizada em diversas plataformas. A limitação dessa abordagem é muito parecida com a abordagem multiplataforma, só é possível desenvolver aplicações para as arquiteturas previamente compatíveis com o framework.

2.1.5 Abordagem: Aplicações Web Progressivas

Essa abordagem tem ganhado muita visibilidade nos últimos anos devido à expansão da WEB 3.0 onde todas as informações na web são usadas de forma mais inteligente e eficiente. Aplicações Web Progressivas (PWA's) são aplicações que levam os conceitos web porem estendem seus domínios à utilização de componentes nativos dos dispositivos. Como se trata de uma aplicação web alguns conceitos originais são mantidos como hospedagem e disponibilização por servidores web, sua instalação também deve ser feita via navegador de internet solicitando a um URL. Para se alcançar recursos visuais semelhantes aos nativos é possível a utilização de HTML e CSS assim como na abordagem hibrida.

A vantagem dessa abordagem é a possibilidade de construir aplicações que podem funcionar perfeitamente em diversas plataformas que tem suporte a tecnologias web e facilidade no desenvolvimento já que, as tecnologias de desenvolvimento já são bem difundidas no mercado. A desvantagem desse tipo de construção fica por conta das limitações em se utilizar os recursos nativos dos dispositivos, pois, elas rodam a partir

dos browsers e tem a limitações de disponibilização, essas aplicações não podem ser disponibilizadas em plataformas oficiais e só podem ser instaladas mediante a URL.

2.2 Ferramentas

Foi apresentado anteriormente as principais metodologias utilizadas na criação de aplicações multiplataformas. Agora será abordado o principiai framework que atualmente está sendo utilizado para a aplicação das abordagens citada, o React Native. React native é bem conhecido e já tem adesão de grandes empresas, suas bibliotecas e documentação estão em constante crescimento.

2.2.1 Framework REACT NATIVE

O framework React Native foi baseado no já consolidado REACT e foi desenvolvido pelo Facebook possibilitando o desenvolvimento mobile podendo disponibilizar simultaneamente aplicações para Android e iOS. Esse ambiente de desenvolvimento foi pensado para se trabalhar com as bibliotecas de JavaScript React.

2.2.2 JavaScript

O JavaScript é uma linguagem de programação script multiplataforma orientada a objetos. Tem com aspectos ser leve e compacta e tende a integrar, de maneira simplificada, outros produtos e aplicações, tendo também a característica de não ser muito bem utilizada sozinha. Em hospedagens, o JavaScript pode servir como ponte que comunica todos os objetos e deve desempenhar o controle sobre a programação entre eles. (MDN, 2021)

Brendan Eich foi o criador da linguagem em 1995 onde foi chamado pela empresa Netscape para criar aplicações que pudessem rodar em seu navegador recém desenvolvido, o Netscape 2.0. O objetivo naquela época era criar uma linguagem leve e interpretada que pudesse complementar o Java que em resumo é uma linguagem complicada e complexa.(SEVERANCE, 2012)

Oficialmente a linguagem JavaScript é chamada de ECMAScript, pois, foi a associação ECMA(European Computer Manufactures Association) quem criou os padrões e especificações da linguagem em 1996. O nome JavaScript já havia sido popularizado pela Oracle (antiga Sun Microsustems) e por isso foi necessário utilizar outra nomenclatura. Oficialmente ficou sedo chamada de ECMAScript (RAUSCHMAYER, 2014). A última versão revisada foi a ECMAScript 2019 popularmente chamada de JavaScript.

2.2.3 Biblioteca REACT

A biblioteca React foi desenvolvida com o proposito de ser eficiente e flexível na criação de interface com usuário. Ela permite ciar interfaces complexas a partir de pequenos e isolados códigos denominados componentes. (REACT, 2021)

A ideia básica é permitir a criação de componentes encapsulados que consegue gerenciar seu próprio estado e criar interfaces complexas a partir de combinações de componentes. Dessa forma é possível criar Views simplificadas representando cada estado, o React irá fazer o gerenciamento de atualização e renderização de forma otimizada apenas dos elementos que sofreram algum tipo de mudança.

Um componente em React é em sua essência uma classe JavaScript que estende a classe "React.Component" e implementa um método de renderização. Esse método é responsável por retornar o que será exibido para o usuário através de sintaxe JSX. Componentes também podem dar origem a função, sendo conhecido mais popularmente com "componentes de função". JSX é uma extensão de sintaxe JavaScript e descreve como a UI será apresentada, sendo possível visualizar a logica da renderização e mais informações dos objetos que serão executados. O JSX não é obrigatório na utilização do Framework React Native porem é a maneira que os desenvolvedores recomendam para utilizar essa biblioteca. A adoção da padronização já existe também é muito recomendada. A Figura 1 demonstra o exemplo de um componente usando o método de renderização 'render()'.

Figura 1 – Componente Usando React

```
class HelloMessage extends React.Component {
  render() {
    return (
      <div>
        Olá, {this.props.name}!
      </div>
    );
  }
}
```

Fonte – REACT (2021)

Para conseguir renderizar os elementos JSX contidos nos componentes, o React utiliza um artifício chamado React DOM que é basicamente um mapeamento em JavaScript da estrutura do DOM (Document Object Model) podendo ser chamado de Virtual DOM. Ele faz a otimização na renderização dos elementos fazendo a comparação entre os "nodes" existentes no Virtual DOM e no DOM, renderizando apenas o que for necessário na tela.

Em React existem dois conceitos muito importantes que são "props" e "state", que simbolizam as informações que os componentes possuem, ambos com suas particularidades.

Props são atributos do JSX que podem ser passados para outros componentes que são usados dentro de uma sintaxe JSX, a informação é passada para um componente e pode ser acessado dentro de um escopo, outros componentes não podem alterar suas informações. Para poder manipular seus próprios dados, um componente usa o conceito "state", sendo ele um objeto privado e gerenciado pelo componente.

Os componentes em React tem um ciclo de vida que possuem métodos que, em momentos específicos, executam trechos de código para uma determinada função. Normalmente os métodos chamados são: constructor(), render(), componentDidMount(), componentDidUpdate() e componentWillUnmount().

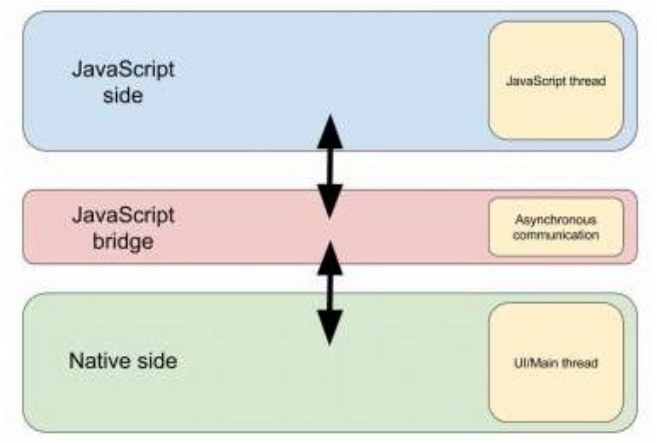
2.2.4 REACT NATIVE

O React Native é um framework utilizado para a criação de aplicativos multiplataformas e utilizada a abordagem de desenvolvimento interpretada como descrito em seções anteriores. A linguagem de programação utilizada é o JavaScript, o código gerado é interpretado pela engine do JavaScript presente nos dispositivos para conseguir acesso as API's e para ter acesso a renderizações nativas.

O React tem papel fundamental na criação de interfaces de usuário e por esse motivo os conceitos introduzidos anteriormente podem ser aplicados aqui, tendo como diferença que no React Native os componentes "View", "Text" e "Image" são mapeados para serem construídos direto pelos componentes de criação de UI nativos da plataforma.

O React Native troca informações com as APIs dos dispositivos através de uma ponte JavaScript utilizando o JavaScriptCore como interpretador. A Figura 2 realizará uma visualização dessa comunicação.

Figura 2 – Comunicação do React Native com API's nativas da plataforma.



Fonte – Hansson e Vidhall (2016)

Todas as solicitações em JavaScript são feitas na ponte, ela irá tratar individualmente cada uma para se adequarem aos tipos nativos das plataformas, no caso de uma solicitação não poder ser tratada imediatamente, essa fica aguardando sua vez em uma pilha de requisições. A interface de usuário nativa não sofre influência durante o processamento feito pela ponte, pois, rodam em threads secundarias e não nas principais como os componentes nativos. (HANSSON; VIDHALL, 2016)

2.3 Método de Avaliação de Arquitetura por Trade-off (ATAM)

O método de avaliação ATAM permite uma análise transacional e consecutiva. Dessa forma é possível levantar requisitos importantes mais rápido. Com esse método também é possível identificar de forma precoce alguns problemas que se tornam muito mais fáceis de resolver nos estágios iniciais do desenvolvimento.

O ATAM é dividido basicamente em 9 etapas(Tabela 2) onde estão agrupadas por Apresentação, Cenário da Arquitetura, Analise e DOC. O primeiro grupo introduz o método com apresentação de objetivos e as soluções arquiteturais respectivas. A etapa 4 identifica basicamente as abordagens relacionadas a qualidade do sistema, a 5 etapa já lida com a criação de uma árvore utilidade (AU) onde é abordado os direcionamentos do negócio em metas de qualidade e cenários concretos que representam os objetivos.

As etapas finais do método identificam as decisões arquiteturais chaves para o cumprimento dos atributos de qualidade, os pontos de trade-offs e os riscos.(COLQUITT; LEANEY, 2007)

2.3.1 Plano de evolução utilizando o ATAM

No modelo ATAM são utilizados 9 passos como método de avaliação. O resultado gerado por essa aplicação prática é o detalhamento, em relatórios, dos novos requisitos de qualidade, os riscos, não-riscos, pontos criativos, pontos de conflitos, monitoramento e outros. Os dados levantados no ATAM serão analisados levando em conta cada ponto de vista do modelo de referência ODP (Modelo de Referência para Processamento Distribuído Aberto). O plano de Evolução será gerado a partir dos resultados alcançados ou observados em cada etapa do modelo de referência. Abaixo será descrito detalhadamente o que é feito em cada etapa do modelo ATAM.

2.3.2 Etapa 1 - Apresentação

A primeira etapa desse processo tem como finalidade promover a troca de informações entre os participantes. Isso forçará todos os envolvidos a conhecer a arquitetura que será avaliada posteriormente.

Tabela 2 – Passos do ATAM

	ETAPAS	RESUTADO	DESCRIÇÃO
APRESENTAÇÃO	1 - APRESENTAR O ATAM	Esclarecer aos stakeholders	Neste passo é realizada uma reunião onde é descrito o método para os participantes do projeto (*stakeholders*), apresentando as expectativas da avaliação, aprender sobre os as metas de qualidades principais para o sistema e conhecer junto ao arquiteto a arquitetura inicial do negócio e os principais cenários.
	2 - APRESENTAR BUSINESS DRIVERS	Especialista apresenta situação crítica de negócio	Neste passo começa o ATAM propriamente dito. São reunidos os *stakeholders* mais importantes do sistema para facilitar a geração de idéias de cenários de usuários, falhas, e antecipar mudanças.
	3 - APRESENTAR ARQUITETURA	High-level alinhado com plano de negócio	A arquitetura é apresentada em detalhes e os cenários mais importantes e ilustrativos são traçados sobre a arquitetura, ajudando a entender o sistema e, em particular, como dados e fluxos são controlados. Os analistas tentam identificar e sondar as ramificações de estilos arquitetônicos aqui.
	4 - IDENTIFICAR ABORDAGEM ARQUITETURAL	Soluções diferentes são destacadas e discutidas	É utilizado um conjunto de padrões de qualidade e questões de atributos específicos para garantir que os requisitos de qualidade são atendidos pelos cenários. Em particular nós observamos se as condições limites tenham sido consideradas
CENARIO ARQ.	*5* - GERAR ÁRVORE DE UTILIDADE.	Core business x requisitos técnicos mapa cenário	Os *stakeholders* votam nos cenários que acham mais importantes. Durante esta fase eles podem sugerir agrupamentos de cenários. Depois que a votação esta completa, os avaliadores determinam um ponto de corte com até 15 cenários.
ANALISE	6 – ANÁLISE E ELEMENTOS ARQUITETURA	Cada cenário é analisado e priorizado Criticidade Sensibilidade	Neste passo o arquiteto observa cada atributo de qualidade estratégico dos cenários especificados, mostrando como eles afetam a arquitetura e como a arquitetura responde a esses atributos (por exemplo, para atributos como desempenho, segurança e disponibilidade).
	7 – BRAINSTORM PARA PRIORIZAÇÃO DE CENÁRIOS	Grupo de stakeholders Expansão de cenários	O arquiteto guia a analise mostrando porque a arquitetura atende a os requisitos do atributo, como iluminado pelo cenário de interesse. O analista constrói modelos de cada atributo de qualidade baseado nas informações do arquiteto.
	8 – ANALISA SOLUÇÕES ARQUITETURA	Repete item 6	Para achar os pontos de conflitos é necessário localizar todos os elementos arquiteturais importantes onde existem múltiplos pontos sensitivos.
DOC	*9* – FORNECEM DOCUMENTOS PARA STAKEHOLDERS	Comunicação	Esse plano é um conjunto de recomendações para reestruturar a arquitetura sob a luz dos resultados da analise. Adicionalmente devem ser levantadas questões sobre a documentação como: informação arquitetural, cenários, informação ambiental, detalhes de restrições e justificativas para os requisitos de qualidade.

Fonte – Colquitt e Leaney (2007)

2.3.2.1 Passo 1 - Apresentar o ATAM

O avaliador chefe deve descrever o método de avaliação (Figura 3) para os participantes, esclarecer os objetivos, as atividades desenvolvidas e as saídas, que pode ser feito por comunicado, publicação ou reuniões. (CLEMENTS; KAZMAN; KLEIN, 2009)

No caso de não ser possível realizar uma reunião a reunião deverá ser feita em um documento explicando o processo, todos envolvidos e suas respectivas obrigações.

Figura 3 – Esquema: Apresentação do ATAM

Fonte – Clements, Kazman e Klein (2009)

2.3.2.2 Passo 2 – Apresentação dos objetivos de negócio

Nesse passo é descrito os objetivos de negócio que são foco da equipe de avaliadores e desenvolvedores, resultando assim nas características de qualidade de interesse do sistema. A Figura 4 é o resumo desse passo onde é processado a documentação e características do sistema, restrições e as saídas. Esse material é essencial para o desenvolvimento das próximas etapas. (CLEMENTS; KAZMAN; KLEIN, 2009)

É importante ressaltar que na saída os objetivos de negócios devem ser muito bem retratados através de cenários que representam os objetivos dos stakeholders.

Figura 4 – Esquema: Apresentação dos objetivos de negócio

Fonte – Clements, Kazman e Klein (2009)

2.3.2.3 Passo 3 – Apresentação da arquitetura

Esse passo é dedicado para o arquiteto dar foco no modo com o qual é feito o atendimento dos objetivos propostos. É sugerido 3 passos: entrada, atividades desenvolvidas e as saídas. São requisitos para execução das atividades o escopo do sistema, lista com objetivos de negócio e a lista de atributos de qualidade. O que é esperado desse passo é a documentação mais completa da arquitetura que pode ser interpretada ou representada por diversas visões ODP com o objetivo de facilitar a visualização dos elementos que serão implementados.(CLEMENTS; KAZMAN; KLEIN, 2009)

2.3.3 Etapa 2 – Investigação e análise

O objetivo principal é realizar o estudo completo da arquitetura e fazer o levantamento de todas as necessidades.

2.3.3.1 Passo 4 – Identificando métodos arquiteturais

Aqui será feito o estudo da documentação gerada nos passos anteriores com o propósito de extrair elementos da arquitetura vigente. Será realizada a verificação da cobertura de cenários tendo como resultado a documentação dos métodos e estilos encontrados.

2.3.3.2 Passo 5 – Gerar arvore de utilidade

O passo 5 fica reservado para a criação de AU[1] baseando-se nos cenários e documentos gerados nos passos 2 e 3. Os atributos de qualidade aqui serão extraídos para serem modelados nos próximos passos.

É de suma importância que todos os requisitos identificados contenham informações complementares que possibilitem aferir se os requisitos requeridos estão sendo de fato implementados ou avaliados.

Caso já exista uma árvore de utilidade primária a mesma pode ser utilizada como uma entrada no passo. A resulta é a criação da documentação da árvore de utilidade elaborada por avaliadores como mostrado na Figura 5.

2.3.3.3 Passo 6 – Análise dos Métodos arquiteturais

Nesse passo é avaliado todo material adquirido nos passos anteriores tendo como foco os resultados do passo 1, 2 e 3 (levantamento de requisitos). A abordagem aqui foca na distinção dos requisitos que terão implementação parcial, totalmente ou não serão atendimentos.

[1] AU - Árvore de Utilidade

Figura 5 – Esquema Gerar Arvore de Utilidade

Fonte – Clements, Kazman e Klein (2009)

2.3.4 Etapa 3 – Teste

Essa etapa compreende o estudo da arquitetura bem como desenvolver cenários para testar se houve o cumprimento das necessidades e requisitos inicialmente planejados e apurados nas etapas anteriores.

2.3.4.1 Passo 7 – Brainstrorm e polarização de cenários

Dentre todos os passos esse pode ser considerado o mais importante de todos pois ele sugere uma brainstorm que reúne todos os stakeholders com a finalidade de encontrar cenários adicionais. Após isso é realizada a análise dos mecanismos arquiteturais a fim de visualizar os senários semelhantes ou conflitantes. Essa visualização pode ser representada como um mapa e pode ser extraído das AU geradas no passo anterior. Isso possibilitará agrupar os cenários semelhante afim de diminuir a AU principal. O próximo passo será a polarização que tem sua prioridade eleita por votação.

A polarização serve para categorizar os cenários de acordo com os atributos de qualidade que estão relacionados e classificados de acordo com sua importância e complexidade.

Normalmente os cenários que recebem prioridade máxima podem forçar os envolvidos a tomar decisões devido a possibilidade de conflitos surgirem com os demais. A figura 6 demonstra o esquema desse passo.

2.3.4.2 Passo 8 – Análise dos métodos arquiteturais

Aqui nesse passo os resultados obtidos no passo 6 são reafirmados e os cenários prioritários gerados no passo anterior são fundamentais para se alcançar uma análise mais apurada.

Figura 6 – Brainstorm e polarização de cenários

Fonte – Clements, Kazman e Klein (2009)

De acordo com Clements, Kazman e Klein (2009), a análise proposta é realizado através de templetes que demonstram os estímulos, respostas e abordagens em cada cenário de alta prioridade definida anteriormente.

2.3.5 Etapa 4 – Entrega

Aqui é a fase final do processo de avaliação segundo o modelo ATAM.

2.3.5.1 Passo 9 – Consolidar os Resultados

Fase na qual são gerados relatórios contendo todas as informações coletadas, as análises e as decisões. Esses relatórios são entregues para todos os envolvidos na evolução do software.

Esse relatório contém uma análise arquitetural dos cenários com maior prioridade, pontos críticos e os trade-offs. Os principais pontos resultantes do ATAM para cumprir com a evolução do software são:

- Conjunto de requisitos;
- Conjunto de riscos e não riscos;
- Conjunto de pontos sensitivos e críticos;
- Pontos de conflitos;
- Plano de monitoramento de riscos.

A partir de todos os documentos gerados é feito o relatório de estratégia de evolução do software que pode ser avaliado e auxilia em qual linha ou cenário a evolução irá ocorrer.

Essa analise é realizada pelos envolvidos e normalmente priorizam os pontos sensitivos e os trade-offs.

Os riscos, não-riscos, pontos sensitivos e trade-offs estão associados sempre a uma árvore de utilidade e passam por refinamento dos atributos de qualidade. Vale ressalta que especificamente os riscos e não-riscos devem conter um plano de monitoramento de riscos vinculados.

3 Desenvolvimento

O foco desse capitulo será realizar a aplicação das metodologias discutidas anteriormente na implantação de melhorias no Software Ufitness que tem como objetivo auxiliar na montagem de dietas mais saudáveis e trazer mais informações em relação aos alimentos consumidos.

O software Ufitness foi lançado oficialmente em 11 de junho de 2020 e tem como principal objetivo criar um plano de refeições e manter uma na rotina saudável, calcular o gasto calórico para adequar-se aos objetivos de perda e ganho de massa corporal.

No entanto o aplicativo não foi lançado para uso comercial. Teve origem em um estudo realizado com foco nos distúrbios alimentares que tiveram grande aumento nos últimos anos.

Após o lançamento da primeira versão na loja Oficial Play Store foi detectado através de pesquisa e observação a necessidade de implementar novos recursos na aplicação que pudessem facilitar ainda mais sua utilização. Essas mudanças e melhorias são os objetos de estudo desse trabalho.

3.1 Avaliação

Foi então desenvolvido um planejamento de evolução do software para que seja possível adequar as novas funcionalidades sem o comprometimento do software e de suas características originais.

Participantes da Avaliação:

Desenvolvedor 1 – Foi quem deu origem ao projeto e assumiu o papel de desenvolvedor e pesquisador na fase inicial

Desenvolvedor 2 – Entrou no projeto para auxiliar no desenvolvimento de novas funcionalidades e se encarregar de futuras pesquisas

Clientes – São os usuários a aplicação da loja PlayStore (A aplicação foi desenvolvida inicialmente para Android)

Orientadores – São as pessoas que acompanham o processo de desenvolvimento da aplicação contribuindo com sugestões de melhorias.

3.1.1 Início da Avaliação

Para realizar a avaliação das funcionalidades e melhorias que serão implementadas no sistema, será utilizado uma versão simplificada do processo ATAM descrito anteriormente no Capítulo 2, tirando o conceito de geração excessiva de documentos e focando nas possíveis implementações práticas. Como se trata de poucas mudanças e o aplicativo ainda não ter uma vasta gama de clientes, irei recolher informações dos comentário e opiniões recebidas em relação ao aplicativo que já está disponível no mercado.

Inicialmente a avaliação da evolução foi proposta por sugestões de Orientadores e posteriormente foi feita uma reunião entre Desenvolvedor 1 e Desenvolvedor 2 de modo a esclarecer os pontos mais importantes dessa proposta de melhoria. Em seguida foi dado início a identificação dos objetivos do negócio para seguir com as melhorias propostas e dando foco nos clientes que são uma ótima fonte de requisitos.

3.1.2 Cenários do Usuário

Após essas atividades iniciais terem sido concluídas foi gerado um quadro, representado na Tabela 3, contendo os requisitos levantados pelos clientes:

Tabela 3 – Cenários do Usuário

U01: Sistema deve permitir a montagem de dietas e cardápios
U02: O sistema deve permitir salvar refeições para serem aproveitadas
U03: O sistema deve permitir criar alimentos para compor as refeições
U04: Sistema deve permitir calcular a necessidade calórica mediante a informaçoes básicas como peso, altura, sexo e idade
U05: Sistema deve mostrar de forma simplificada os resultados adquiridos com os dados inseridos
U06: Sistma deve permitir consultar as informações nutricionais de alimentos. Ex: Uso da tabela Taco

Fonte – Do Autor.

3.1.3 Cenários de Negócio

No quadro da Tabela 3 estão as principais características que são importantes para usuários do aplicativo. A maioria desses requisitos já estão implantados ficando assim o foco principal na implantação de uma forma de consultar alimentos que não foram cadastrados, que possam ser fornecidos de outras fontes, como a tabela TACO e a tabela Guilherme Franco.

Após avaliar as necessidades principais dos clientes foi necessário fazer o levantamento dos cenários referentes aos atributos não funcionais.

Fazendo o levantamento das necessidades técnicas é possível identificar os principais requisitos que o sistema deve abranger para que a evolução tenha efeito positivo. Aqui

Tabela 4 – Cenários de Negócio

N01: O sistema deve ser executado em qualquer dispositivo Android 4.4 ou superior
N02: O sistema não deve comprometer o desempenho dos dispositivos durante sua utilização
N03: Sistema deve ser de facil instalação
N04 Sistema deve permitir a implantação de novas funcionalidades sem comprometer as que já existem
N05: Sistema deverá ser desenvolvido em JavaScript utilizando o ReactNative
N06: Sistema deve garantir acesso aos dados a qualquer momento
N07: Sistema deve garantir performace quando estiver fazendo requisicoes externas. Consumo de API

Fonte – Do Autor.

não há reprovação de requisitos, todos são aceitos de modo a comporem uma avaliação mais rigorosa nos próximos passos. A Tabela 4 demonstra os resultados dessa fase.

3.1.4 Resumo da Fase 3

Na Tabela 5 são levantados os pontos de vista dos desenvolvedores, usuários e tecnologias empregadas na versão atual do sistema para que seja possível abordar possíveis pontos sensíveis do projeto.

Tabela 5 – Resumo da Fase 3

Visão	Artefato
Ponto de vista de Empresa	Entregar um software funcional
Ponto de vista Tecnologia	Usar o React Native como forma principal de Desenvolvimento
Ponto de vista de Informação	Processar informações nutricionais de acordos com regras passadas previamente
ponto de vista de Engenharia	Arquitetura principal: Android e iOS
Ponto de vista de Computação	Entregar um software funcional

Fonte – Do Autor.

3.1.5 Abordagem Arquitetural

Em seguida são identificados os métodos arquiteturais, utilizando o ATAM fica muito mais fácil identificar as abordagens e estilos utilizados na implementação de atributos de qualidade que tem prioridade.

Métodos arquiteturais identificam e descrevem as formas com a qual um sistema pode crescer, adaptar a possíveis mudanças e se for o caso interagir com outros sistemas e arquiteturas.

A segui um quadro, Tabela 6, expondo o levantamento da abordagem arquitetural identificada o Software Ufiness. Como observado manutenibilidade e segurança dos dados,

como os componentes interagem entre si, e uma descrição dos benefícios da utilização desse estilo.

Tabela 6 – Abordagem Arquitetural

Abordagem Arquitetural
O sistema utiliza componentização
O software utiliza as bibliotecas padrões de inteface de acordo com cada dispositivo
O sistema permite mudanças por meio da componentização
Sistema permite armazenamento dos dados de forma permanente com a utilização de banco de dados
Sistema permite consumo de API externa
Sistema permite funcionamento parcial sem utilização da Internet
Sistema permite manutenção facilitada devido a arquitetura do Framework de desenvolvimento

Fonte – Do Autor.

3.1.6 Arvore de Utilidade

O próximo passo consiste na montagem de uma árvore de utilidade, Tabela 7, que tem a função de agrupar e priorizar os requisitos levantados anteriormente. Aqui foram adicionados novos requisitos e alguns foram detalhados de forma que facilite o entendimento dos desenvolvedores.

3.1.7 Requisitos E Mecanismos Existentes

O sexto passo tem como objetivo verificar se os requisitos levantados na árvore de utilidade estão sendo contemplados pela arquitetura atual do sistema.

Para que novas implementações possam surgir é de suma importância o conhecimento prévio dos métodos utilizados no projeto atual para que seja possível analisar as próximas abordagens (Tabela 8).

3.1.8 Finalizando o ATAM

Os próximos passos que serão estudados terão como objetivo analisar a viabilidade dos requisitos levantados e inserir nesse contexto as mudanças que serão de fato implementadas tendo assim o desvio do processo ATAM já que o mesmo é focado na criação da documentação e não no desenvolvimento em si.

Após entender melhor as necessidades dos clientes e dos envolvido foi foram coletadas mais opiniões de maneira informal sobre as possíveis melhorias e mudanças que poderiam ser implementadas de modo a agregar no software inicial, funções interessantes que não comprometessem a proposta original.

O resultado disso foi que existe uma necessidade de se facilitar o processo de criação de alimentos uma vez que nem sempre temos as informações nutricionais necessárias para que os cálculos possam ser realizados de forma coerente.

Tendo em vista os resultados demonstrados na tabela "Requisitos E Mecanismos Existentes", Tablela 8, foi possível analisar que o sistema ainda não está preparado para realizar requisições de informações externas e por esse motivo alternativas foram pensadas de modo a resolver esse problema.

Tabela 7 – Arvore de Utilidade

Funcionaliadde	Adequação	O sistema deve disponibilizar todas as funçoes da versão anterior.
	Precisão	Garantir o melhor calculo calórico dos nutrientes e das refeições
Confiabiliade	Tolerancia a Falhas	Sistema deve tratar as informaões fornecidas pelo usuario afim de previnir erros nas medições e cálculos
	Recuperabilidade	Em caso de falhas, o sistema deve permitir o retorno sem perda de informações esseciais
Usabilidade	Compreensibilidade	O sistema deve ser intuitivo e de facil utilização, os usuarios devem conseguir utilizar de forma plena o aplicativo com pouco mais de 30 minutos.
Eficiencia	Desempenho	Os dispositivos não podem ficar lentos durante a utilização do sistem. As consultas não podem apresentar demora na apresentação dos resultados
Manutenibilidade	Analisabilidade	O sistema deve permitir realizar uma analise dos resultados inseridos no programa
	Modificabilidade	O sistema deve permitir a implantação de novas funcionalidades
		O sisetma deve permitir a expanção para ourtas plataformas
	Testabilidade	O sistema deve permitir testar funcionalidades sem o comprometimentos dos dados arquivados
Portabilidade	Adaptabilidade	Sistema deverá ser instalado pela Loja oficial de aplicativos da plataforma em Android 4.4 ou superior ou em iOS
		Sistema deve utilizar Banco de Dados RealmDB
	Instabilidade	Sistema deve ser instalado rápido e não pode comprometer a utilização dos dispositivos durante o processo.
	Co-existencia	Sistema deve adquirir informações de forma externa sem o comprometimento da arquitetura já desenvolvida
Atributos de Negócio	Time-to-market	Novas versões devem ser disponibilizadas de forma fácil, coerente e rápidas
	Custo e benefício	O sistema deve ser construído de modo a facilitar possiveis integrações de pessoal na equipe de desenvolvimento, diminuindo custo e tempo em desenvolvimentos futuros

Fonte – Do Autor.

Tabela 8 – Requisitos E Mecanismos Existentes

Requisitos Levantados	Abordagem
O sistema deve disponibilizar todas as funçoes da versão anterior.	ATENDE: Sistema se caracteriza pelo uincremento de função
Garantir o melhor calculo calórico dos nutrientes e das refeições	Atende: Sistema baseia os cálculos em documentos oficiais.
Sistema deve tratar as informaões fornecidas pelo usuario afim de previnir erros nas medições e cálculos	ATENDE PARCIALMETNE: Nos tratametnos das informações nem todos os cenários foram abordados.
Em caso de falhas, o sistema deve permitir o retorno sem perda de informações esseciais	NÃO ANTENDE: Sistema não implementa nenhum recurso específico para recuperação de dados em caso de falhas.
O sistema deve ser intuitivo e de facil utilização, os usuarios devem conseguir utilizar de forma plena o aplicativo com pouco mais de 30 minutos.	ATENDE: Sistema muito simples, intuitivo e conta com uma sessão de ajuda.
Os dispositivos não podem ficar lentos durante a utilização do sistem.	ATENDE: Durante a utilização não foi observado nenhum trabamento ou perda de desempenho no sistema
As consultas não podem apresentar demora na apresentação dos resultados	ATENDE PARCIALMENTE: Ainda não foi testado com uma quantidade massiva de dados em ambiente real.
O sistema deve permitir realizar uma analise dos resultados inseridos no programa	ATENDE: Existe a forma de visualizar os resultados montados em um gráfico utilizando a função ACOMPANHAMENTO
O sistema deve permitir a implantação de novas funcionalidades	ATENDE: Com as facicildades encontradas no Framework é possivel implementar novas funcionalidades no sistema.
O sisetma deve permitir a expanção para ourtas plataformas	ATENTE PARCIALMENTE: O Framework é limitados para plataformas expecíficas.
O sistema deve permitir testar funcionalidades sem o comprometimentos dos dados arquivados	ATENDE: Sistema atualmetne permite o cadastro de alimentos erefeições e sua exclusão para o caso de realizar testes
Sistema deverá ser instalado pela Loja oficial de aplicativos da plataforma em Android 4.4 ou superior ou em iOS	ATENTE PARCIALMENTE: Está disponível para Android e não conta com versão para iOS ainda.
Sistema deve utilizar Banco de Dados RealmDB	ATENDE: Banco de dados utilizado Realm DB por sua compatibilidade mobile.
Sistema deve ser instalado rápido e não pode comprometer a utilização dos dispositivos durante o processo.	ATENDE: Processo de intalação simulado em alguns dispositivos com Hardwares diferentes e nenhum apresentou problemas
Sistema deve adquirir informações de forma externa sem o comprometimento da arquitetura já desenvolvida	NÃO ATENDE: Atualmetne o sistema não faz requisições externas.
Novas versões devem ser disponibilizadas de forma fácil, coerente e rápidas	ATENDE PARCIALMENTE: Existe apenas uma versão divulgada do aplicativo.
O sistema deve ser construído de modo a facilitar possiveis integrações de pessoal na equipe de desenvolvimento, diminuindo custo e tempo em desenvolvimentos futuros	ATENDE PARCIALMENTE: Apesar de tem um código limpo e de fácil compreenção não existe uma documentação Oficial

Fonte – Do Autor.

3.1.9 Resultados das Análises

Para facilitar o cadastro de alimentos foi pensado na integração com API ou arquivos contendo um volume expressivo de informações previamente cadastradas. Das duas abordagens verificou-se que se for utilizado um documento com todas as informações úteis ao sistema, o processamento dos dados poderia comprometer a integridade e instabilidade

do aplicativo. Foi então utilizado para obter os registros dos alimentos uma API contendo as informações nutricionais dos alimentos. Essa abordagem é muito útil, pois, não há a necessidade de realizar o tratamento de arquivos brutos, fazendo assim a aplicação não ter que gastar recursos processando arquivos muito grandes nem instruções complexas.

Durante o levantamento de ideias também surgiu algumas opiniões referentes às mudanças nos layouts para gerar uma inovação visual na aplicação. Essa solicitação tem referência às tendências do mercado onde as aplicações quando trazem novas funcionalidades acabam trazendo novidades visuais mesmo que pequenas como fontes, cores, arredondamentos ou até mesmo a reestruturação das informações dispostas na tela.

Outra mudança que surgiu foi a possibilidade de trazer receitas como um adicional. Essas receitas tem o foco de dar algumas opções saudáveis e fáceis de serem feitas. Essa está em processo de análise.

Nas próximas seções e capítulos serão apresentados os resultados de toda a análise de mudança e possíveis problemas durante as implementações.

3.2 API TACO

A tabela TACO[1] é um projeto que se originou da NEPA (Núcleo de Estudos e Pesquisas em Alimentos) da UICAMP e teve financiamento Oficial do Ministério da Saúde e do Ministério do Desenvolvimento Social e Combate à Pobreza e seu objetivo principal era fornecer informações nutricionais de uma grande quantidade de alimentos nacionais e regionais. Os dados foram adquiridos através de amostragem e os testes forma realizados em laboratórios reconhecidos mundialmente pela qualidade de suas pesquisas.

A API Taco[2] utilizada tem como objetivo principal é adquirir os dados originais da pesquisa e disponibiliza-los para que outros desenvolvedores possam usar as informações de forma fácil, de forma que possa ser integrada a qualquer outra aplicação. Originalmente as informações da Tabela Taco estão disponibilizadas das seguintes formas:

- Por documento de texto PDF: Nesse caso a informação está no formato bruto e as vezes pode confundir quem está pesquisando devido a impossibilidade de aplicar filtros no documento que possam auxiliar na visualização das informações.
- Por arquivo tabulado XLS: Basicamente esse é o formato de planilha muito conhecido atualmente. Dessa forma é possível formatar o documento para impressão ou para extração de dados de acordo com as capacidades de quem está manipulando o documento.

[1] Disponível em <http://www.nepa.unicamp.br/taco/tabela.php?ativo=tabela>
[2] Disponível em: <https://taco-food-api.herokuapp.com>

3.2.1 Construção da API

Na construção da API foi usado o método de tratamento de informações extraídos do arquivo XLS descrito na seguinte sequência: primeiro foi extraído as informações do XLS limpando os estilos, linhas e colunas desnecessárias; Em seguida foi a separação dos alimentos e agrupamento por categorias; foi gerado assim o arquivo CSV separando os registros por vírgula e em seguida gerado o JSON[3].

O arquivo JSON criado contem todas as categorias relacionadas aos Alimentos e para obter os dados é necessário fazer as requisições por categorias (category) ou alimentos (food).

3.2.2 Métodos da API

Na documentação oficial da API os comandos básicos utilizados são (Figura 7):

Solicitando uma categoria Específica

Figura 7 – Exemplo de Solicitação de Categoria

Fonte – API TACO (2021)

O resultado dessa solicitação é basicamente apenas de Strings com as descrições, id e o objeto (categoria) de acordo com o quadro da tabela 9.

Tabela 9 – Atributos de Categoria

Campo	Tipo	Descrição
category	Object	Categoria
id	Number	ID unico da Categoria
description	String	Descrição da Categoria

Fonte – API TACO (2021)

No caso de requerer todas as categorias a solicitação ficaria conforme a Figura 8, nesse caso a resultante seria um vetor contendo todos os objetos (Categoria).

Para solicita um alimento é utilizado uma requisição muito parecida com a requisição anterior, porem passando como solicitação o parâmetro foodId (Figura 9).

[3] JSON. JavaScript Object Notation

Figura 8 – Exemplo de Solicitação de todas as Categorias

Fonte – API TACO (2021)

Figura 9 – Exemplo de solicitação de Alimento específico

Fonte – API TACO (2021)

O resultante dessa consulta é um emaranhado de dados referente aos atributos dos alimentos como categoria, Kcal, cálcio, carboidratos por porção, fibras, etc.

Além desses dados muitos outros são obtidos e servem basicamente para a realização dos cálculos que o aplicativo Ufitness utiliza.

A tabela taco utilizada como referência está na sua quarta edição e a API construída está na sua versão 0.1.0. No capítulo 4 será demonstrado, na prática, o resultado obtido com essa implementação.

4 Ferramentas e Resultados

Este capítulo irá demonstrar as mudanças visuais e funcionais na aplicação e em seguida alguns testes utilizados com foco na usabilidade e por fim será abordado os problemas encontrados na implementação dos novos recursos.

4.1 Ferramentas

Como se trata da atualização e evolução de um software existente, optou-se por utilizar várias das mesmas ferramentas e tecnologias utilizadas no desenvolvimento da versão 1 do aplicativo em questão. São elas: o framework de desenvolvimento mobile multiplataforma React Natine versão 0.60 (mesma versão utilizada no desenvolvimento anterior), e o editor de código fonte Visual Studio Code versão 1.45 desenvolvido e distribuído pela Microsoft.

Os equipamentos e dispositivos de hardware utilizados para o desenvolvimento das funcionalidades adicionais do aplicativo, envolvem um computador que possui 16GB de memória RAM, processador Intel Xeon 2630Lv3 1.8 GHz (16-core), placa gráfica GeForce GTX1060 e armazenamento SSD 480GB. Para os testes foi utilizado um smartphone Android versão 6.0, 4GB de memória RAM, processador, Helio X20 10-core 2.3GHz e tela de 5 polegadas. Um segundo smartphone foi utilizado para realizar algumas comparações testes contendo: Android versão 11, 6GB de memória RAM, processador Snapdragon 720G 8-core e tela de 6,58 polegadas. O resumo dos dispositivos utilizados durante o desenvolvimento pode ser visualizado na Tabela 10

Tabela 10 – Equipamentos Utilizados

Equipamentos Utilizados			
	Computador	**Smartphone 1**	**Smartphone 2**
SO	Windows 10	Andoid 6.0	Android 11
RAM	16 GB	4 GB	6 GB
CPU	Xeon E5-2630LV3	Helio x20	Snapdragon 720G
GPU	GTX 1060	Malli T-880	Adreno 630

Fonte – Do Autor.

Optou-se pela utilização da versão 0.60 do React Native, versão utilizada no desenvolvimento da versão 1 do aplicativo, por motivos de compatibilidade e depreciação de componentes e funcionalidades das versões mais recentes do framework, necessários para o correto funcionamento das funcionalidades do aplicativo.

Optando pela mesma versão, pode-se economizar no tempo de reconstrução de funcionalidades existentes, focando somente na adição de novas porem tendo limitações de recurso implementados nas versões posteriores.

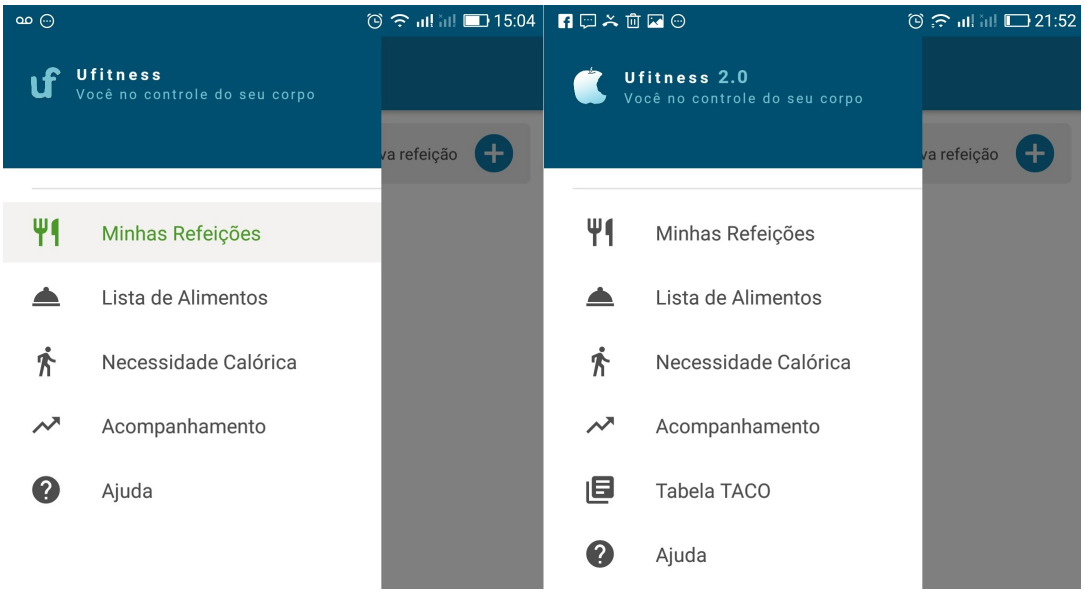
4.2 Menus Adicionados

Com o intuito de manter a proposta minimalista e limpa da versão 1.0, houve pequenas e sutis alterações nos layouts. A aplicação conta com esse menu para guiar o usuário pelas funcionalidades do app.

Ao abrir o menu principal, nota-se que o logo passou por alterações, passando das siglas UF que davam nome ao app, para a figura de uma maçã, o que se adequa melhor a proposta do aplicativo. Além disso, e não menos importante, é possível notar também a adição de uma nova opção no menu principal do aplicativo, a opção "Tabela TACO". Ambas as atualizações podem ser visualizadas na Figura 10.

A opção de menu "Tabela Taco", leva à tela onde o usuário poderá consultar as informações nutricionais dos alimentos presentes na Tabela Brasileira de Composição de Alimentos (TACO). Nela o usuário tem acesso às informações de nome, saldo calórico, carboidratos, proteínas, gorduras, fibras e sódio presentes no alimento buscado.

Figura 10 – Alterações no Menu Principal



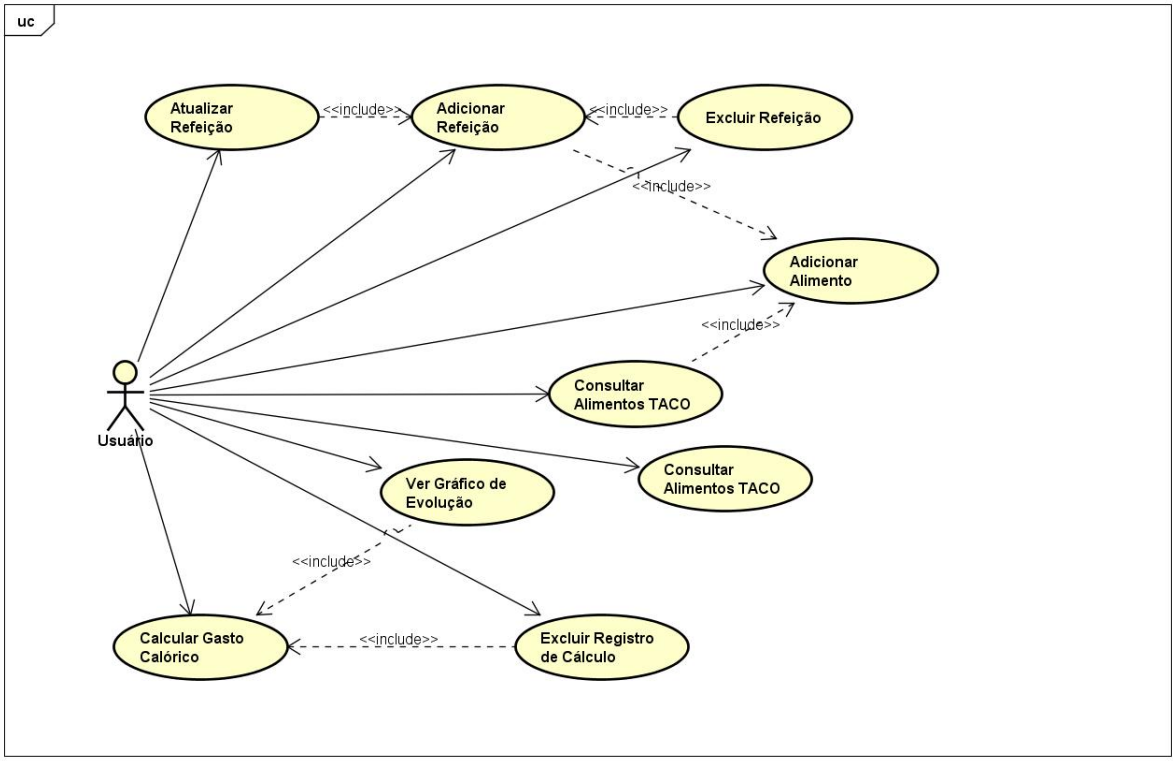
Fonte – Do Autor.

4.3 Descrição de funcionalidades, Telas e Diagramas

Nesta seção será feita uma descrição detalhada sobre as funcionalidades adicionais que foram inseridas na versão 2.0 do app, assim como o planejamento da estrutura dessas funcionalidades.

O Diagrama de Caso de Uso da Figura 11, ilustra o comportamento do usuário dentro do aplicativo de forma geral a partir do menu principal. Ele mostra toda a dependência de funcionalidades que o usuário deve se atentar de modo a executar uma ou outra função.

Figura 11 – Diagrama de Caso de Uso Geral



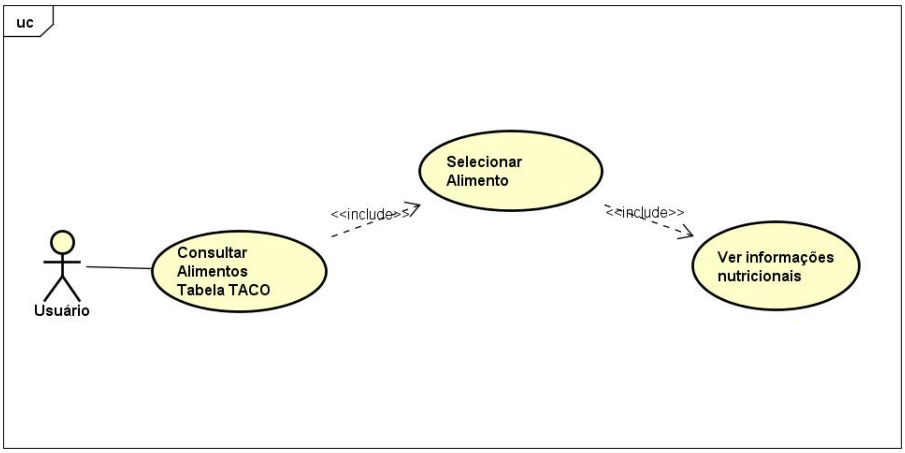
Fonte – Do Autor.

O diagrama de Caso de Uso visualizado na Figura 12, demonstra como o usuário se comporta para consultar um alimento na Tabela Brasileira de Composição de Alimentos (TACO). Para isso, ele deve acessar a opção de menu Tabela TACO do aplicativo, então basta procurar e selecionar o alimento desejado no campo de seleção. Ele poderá ver as informações nutricionais do alimento logo abaixo desse campo de seleção.

4.3.1 Tela Tabela Brasileira de Alimentos (TACO)

Essa tela pode ser acessada através da opção “Tabela TACO” no menu principal. Nela o usuário poderá consultar as informações nutricionais dos alimentos contidos na Tabela Brasileira de Composição de Alimentos (TACO).

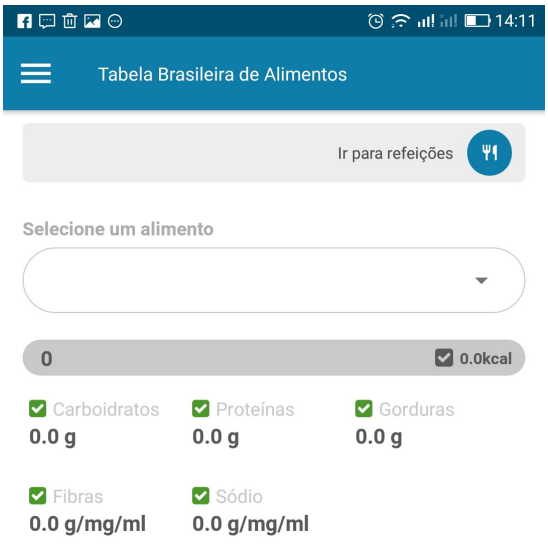
Figura 12 – Diagrama de Consulta Tabela Taco



A tela apresentada na Figura 13 é composta pelos elementos: ícone para o menu principal, o título da janela com o nome Tabela Brasileira de Alimentos e o botão para ir para a tela de refeições. Além disso, existe o campo de seleção, onde é possível o usuário selecionar o alimento do qual deseja saber as informações nutricionais.

Assim que o usuário seleciona o alimento desejado, abaixo, na tela, são exibidas as seguintes informações: nome, valor calórico, carboidratos, proteínas, gorduras, fibras e sódio, contidos no alimento.

Figura 13 – Tela Consulta Tabela Taco



Fonte – Do Autor.

4.3.2 Tela Criar Refeições

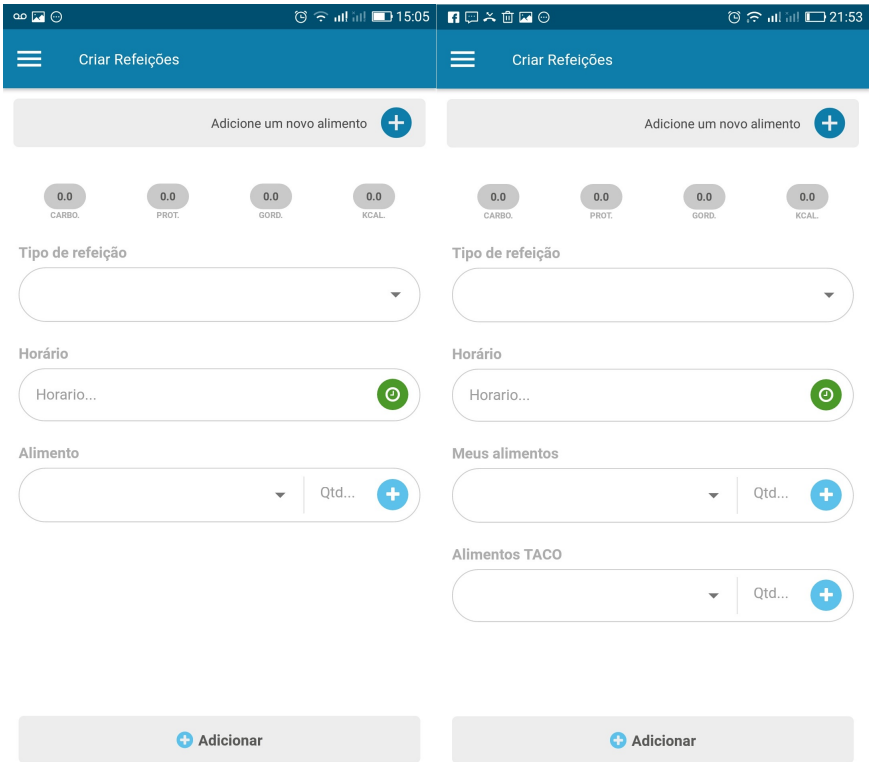
Como na proposta da versão 1.0 do app, essa tela permite ao usuário a adição de novas refeições ao seu plano de dieta. A mesma foi reformulada para que o usuário

tenha agora, a opção de adicionar às refeições, alimentos pré cadastrados, provenientes da Tabela Brasileira de Composição de Alimentos (Taco). A tela pode ser acessada através da opção “Adicionar nova refeição” disposta na margem superior direita da tela “Refeições Agendadas".

Nessa tela, o usuário poderá visualizar os mesmos elementos e opções que na versão anterior do aplicativo. Uma opção para o cadastro de novo alimento no canto superior direito da tela, elementos que marcam o saldo calórico da refeição à medida que novos alimentos são inseridos na mesma e os campos “Tipo de refeição” e “Horário” permanecem os mesmos. O campo de “Alimentos” foi alterado para “Meus alimentos”. Nesse campo o usuário pode selecionar um alimento cadastrado por ele, assim como já o fazia. Por fim, a opção “Alimentos TACO”, que conta com todos os alimentos da tabela brasileira de composição de alimentos, disponível para que o usuário opte por usar para criar suas refeições. O usuário pode inserir tantos alimentos quanto desejar, podendo inclusive mesclar as refeições, com alimentos cadastrados e alimentos da tabela TACO.

Após inserir os dados necessários para a criação da refeição, o usuário deve pressionar o botão “Adicionar” na margem inferior da tela. A tela de cadastro de refeições pode ser visualizada na Figura 14.

Figura 14 – Tela Criar Refeições Antes e Depois



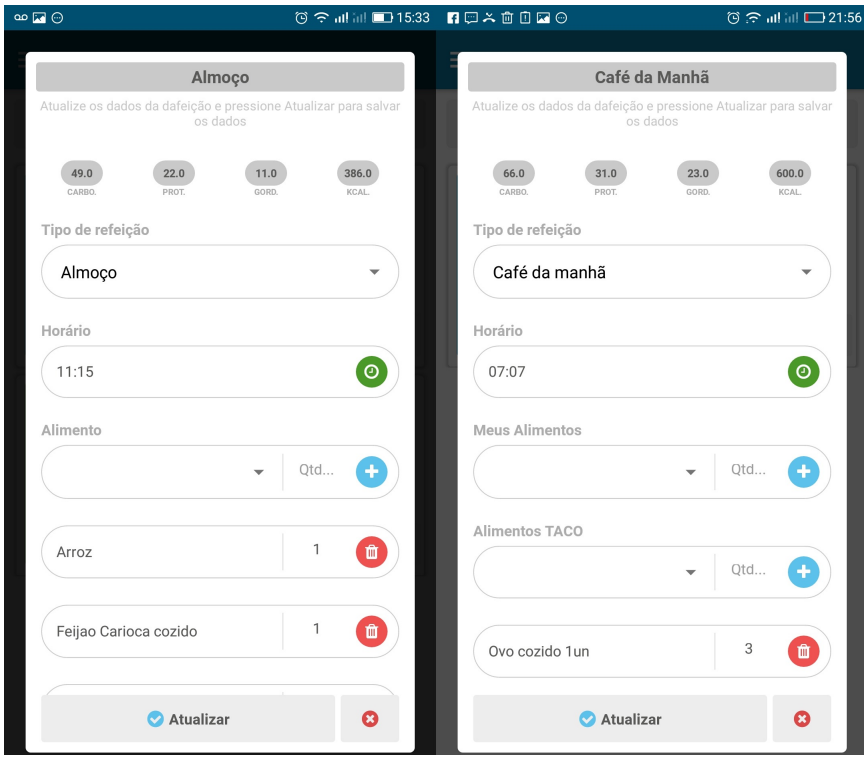
Fonte – Do Autor.

4.3.3 Tela Edição de Refeição

Assim como na tela de cadastro e criação de novas refeições, a tela de edição também foi reformulada para que o usuário tenha a opção de editar os componentes da refeição de acordo com alimentos pré cadastrados, provenientes da Tabela Brasileira de Composição de Alimentos (Taco). O usuário só tem acesso à tela de edição se já existe alguma refeição cadastrada na lista de refeições agendadas na tela principal. Ao clicar no ícone de edição presente no card da refeição o usuário tem acesso à tela para fazer as edições desejadas na refeição escolhida.

Nessa tela representada na Figura 15, o usuário poderá visualizar os mesmos elementos e opções que na tela de criação de refeições reformulada. Nela, o usuário tem a liberdade para a edição das informações sobre o tipo de refeição, horário e exclusão ou adição de novos alimentos à refeição editada. Após as alterações, o usuário tem duas opções: salvar as alterações ou descartar as edições realizadas. Estas duas opções são expressas nos ícones "Atualizar" e "Fechar" dispostos no limite inferior desta tela de edição.

Figura 15 – Tela Edição de Refeição Antes e Depois



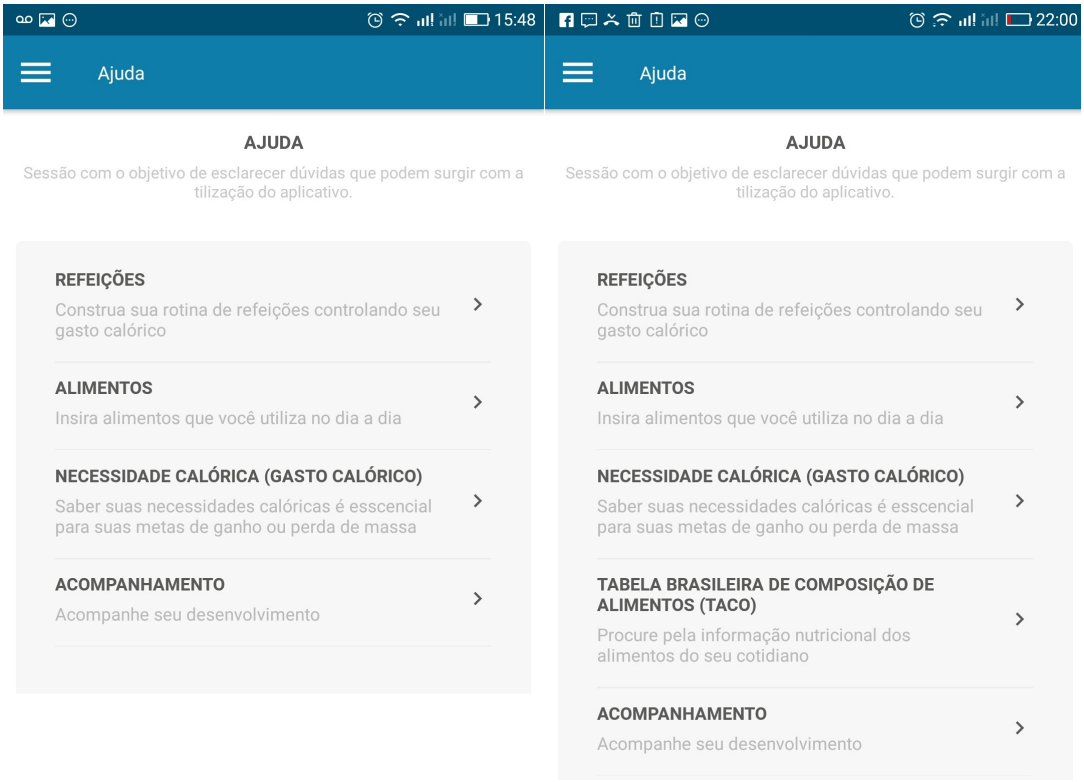
Fonte – Do Autor.

4.3.4 Tela Ajuda

A tela de Ajuda oferece o guia prático para cada uma das funcionalidades do aplicativo. A tela de ajuda pode ser acessada através da opção "Ajuda" no menu principal.

Além dos tópicos já existentes nessa tela, foi adicionado um guia de como o usuário pode fazer consultas dos dados nutricionais na tabela TACO. A comparação entre o antes e depois pode ser verificada na Figura 16.

Figura 16 – Tela Ajuda Antes e Depois



Fonte – Do Autor.

5 Conclusão

Este trabalho propôs demonstrar a evolução de um sistema existente utilizando conceitos já consolidados no mercado para uma boa organização das ideias e das técnicas de desenvolvimento.

Inicialmente foi tratado um estudo sobre a distribuição alimentar da população brasileira de modo a introduzir o tema foco do aplicativo que é criação de dietas e trazer informações úteis para quem tem o objetivo de conhecer melhor suas necessidades calóricas e, possivelmente, emagrecer de forma saudável.

As análises das mudanças foram feitas através de documentação ATAM simplificada para demonstrar em cenário real como seria a programação da evolução do software com foco em melhorar ou implementar novas funções não comprometendo os requisitos de qualidade já existentes no projeto original.

A partir dos cenários levantados e das ideias que foram adquiridas nos passos do ATAM foram implementadas as funções de integrar a criação das refeições com uma API externa que fornece informações atualizadas dos alimentos permitindo também o melhor desempenho na hora de buscar informações externas, já que o aplicativo original não contava com nenhum tipo de requisições de informação externa.

A utilização do React Native foi de grande ajuda na perspectiva de que a aplicação pode ser lançada para outras plataformas sem a necessidade de retrabalho e geração de código extra. O aplicativo foi construído com a proposta de disponibilizar aplicação (multiplataforma) utilizado como base a linguagem de programação JavaScript e possibilitando a utilização de recursos nativos de ambas as plataformas (Android e iOS).

Os problemas encontrados durante o desenvolvimento são basicamente problemas de quebra de código decorrente de atualização no Framework e em algum conceito que, durante o desenvolvimento da atualização a versão estava atualizada e alguns conceitos utilizados no código inicial já não eram mais utilizado, o que forçou a realização de uma análise preliminar do código antes das demais implantações.

O trabalho conseguiu alcançar o objetivo proposto e pretende alcançar um número suficiente de usuários que possam contribuir para o desenvolvimento de novas melhorias e funcionalidades. Os resultados adquiridos com a metodologia ATAM poderão ser base para novos incentivos de implementações.

Referências

API TACO. *DOCUMENTAÇÃO API TACO*. 2021. Disponível em: <https://taco-food-api.herokuapp.com>. Citado 2 vezes nas páginas 41 e 42.

BASS, S. et al. Exercise before puberty may confer residual benefits in bone density in adulthood: Studies in active prepubertal and retired female gymnasts. *J Bone Miner Res*, v. 13, p. 500–507, 1998. Citado na página 17.

BIøRN-HANSEN, A.; GRøNLI; GHINEA, G. T.-M. A survey and taxonomy of core concepts and research challenges in cross-platform mobile development. *Acm Comput. Surv*, p. 1–34, 2018. Citado na página 21.

BOSCH, J. *Design Use of Software Architectures - Adopting and EvoMng a Product LineApproach*. Harlow UK: Addison-Wesley, 2000. Citado na página 17.

CLEMENTS, P.; KAZMAN, R.; KLEIN, M. *Evaluating software architecture: Methods and case studies*. 2009. Citado 5 vezes nas páginas 28, 29, 30, 31 e 32.

COLQUITT; LEANEY. Expanding the view on complexity within the architecture trade-off analysis method. In: *Proceedings of 1he 1481 Annual IEEE lnternational Conference and Workshops on 1he*. [S.l.: s.n.], 2007. Citado 2 vezes nas páginas 27 e 28.

EL-KASSAS; WAFAA, S. Taxonomy of cross-platform mobile applications development approaches. *Ain Shams Engineering Journal*, n. 2, p. 163–190, 2017. Citado 2 vezes nas páginas 20 e 21.

FAO; WHO. Food and agriculture organization/world health organization. necesidades de energia y de proteínas. In: FAO/WHO/UNU. Genebra, 2003. p. 220. Citado na página 14.

FRANCO, G. *Tabela de composição química dos alimentos*. São Paulo: Atheneu, 1992. Citado na página 15.

GODFREY, M.; GERMAN, D. *The past, present, and future of software evolution, In: Frontiers of Software Maintenance*. 2008. 129–138 p. Citado na página 18.

HANSSON, N.; VIDHALL, T. *Effects on performance and usability for cross-platform application development using React Native*. Linköping, Switzerland: [s.n.], 2016. Citado 2 vezes nas páginas 26 e 27.

IBGE. *Pesquisa de orçamentos familiares, 2002-2003. Aquisição alimentar domiciliar per capita, Brasil e grandes regiões*. Rio de Janeiro: [s.n.], 2004. Citado na página 14.

KRUCHTEN, P.; OBBINK, H.; STANFORD, J. *The past, present, and future for software architecture. Software, IEEE*. 2006. 22–30 p. Citado na página 17.

LAGIOU, P.; TRICHOPOULOU, A. The dafne initiative: the methodology for assessing dietary patterns across europe using household budget survey data. *Public Health Nutr*, v. 4, p. 1135–1141, 2001. Citado na página 14.

MDN. *Introduction - JavaScript*. 2021. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Introduction.> Citado na página 24.

MONDINI, L.; MONTEIRO, C. Mudanças no padrão de alimentação na população urbana brasileira. *Rev Saúde Pública*, v. 28, n. 6, p. 433–439, 1994. Citado na página 14.

MONTEIRO, C.; MONDINI, L.; LEVY-COSTA, R. Mudanças na composição e adequação nutricional da dieta familiar nas áreas metropolitanas do brasil (19881996). *Rev Saúde Pública*, v. 34, n. 3, p. 251–258, 2000. Citado na página 14.

PARNAS, D. Software aging. In: *IICSE • 94: Proceedings of the 16th international conference on Software a,gineering*. Los Alamitos, CA, USA: IEEE Computer Society Press, 1994. p. 279–287. Citado na página 18.

PORTES, A. Fundamental laws and assumptions of software maintenance. *Empirical Software Engineering*, Kluwer Academic Publishers, Hingham,USA, n. 2, p. 119–131, 1997. Citado na página 17.

RAUSCHMAYER, A. *Speaking JavaScript: an in-depth guide for programmers*. Sebastopol, CA: O'Reilly Media, Inc, 2014. Citado na página 24.

REACT. *DOCUMENTAÇÃO React*. 2021. Disponível em: <https://reactjs.org/>. Citado na página 25.

SERRA, L. et al. Comparative analysis of nutrition data from national, household, and individual levels: results from a who-cindi collaborative project. In: . Canada, Finland, Poland, and Spain: [s.n.], 2003. p. 57:74–80. Citado na página 14.

SEVERANCE, C. Javascript: Designing a language in 10 days. *Computer*, n. 2, p. 7–8, 2012. Citado na página 24.

SOMMERVILLE, I. *Engenharia de software*. [S.l.]: Pearson Prentice Hall, 2011. Google-Books-ID: H4u5ygAACAAJ. ISBN 9788579361081. Citado 2 vezes nas páginas 18 e 20.

SVAHNBERG, M. Supporting software architecture evolution. In: . Blekinge lnstitute of Technology ,Suécia: Tese(doutorado), 2003. Citado na página 17.

TU, Q.; GODFREY, M. An lntegrated approach for studying architectural evolution. In: _____. *In: 10th lnternational Workshop on Program Comprehension*. Paris, France: [s.n.], 2002. p. 127. Citado na página 18.

Universidade Estadual de Campinas [NEPA/ Unicamp]. Tabela brasileira de composição de alimentos [taco]. In: ALIMENTAçãO, N. de Estudos e Pesquisas em (Ed.). São Paulo; NEPA/ Unicamp: [s.n.], 2004. v. 1. Citado na página 15.