DESENVOLVIMENTO DE UM APLICATIVO MULTIPLATAFORMA PARA PRECIFICAÇÃO DE RECEITAS PARA PEQUENOS RESTAURANTES

Lucas Vidotto[1], Rodrigo de Oliveira Plotze[1]

[1]Faculdade de Tecnologia de FATEC Ribeirão Preto (FATEC)

Ribeirão Preto, SP – Brasil

lucas.vidotto@fatec.sp.gov.br,
rodrigo.plotze@fatec.sp.gov.br

Resumo. *Este artigo descreve o desenvolvimento de um aplicativo destinado a precificar receitas culinárias para pequenos restaurantes. O objetivo é facilitar a gestão do lucro das receitas, seguindo as abordagens descritas dentro da teoria cost-price, com foco nos métodos Cost-Based Pricing e Break-Even Pricing, utilizada de maneira abrangente em todos os setores gerenciais.*

Abstract. *This article describes the development of an app to price culinary recipes for small restaurants. The objective is to facilitate the profit management of recipes following the approaches described within the cost-price theory with focus on method Cost-Based Pricing and Break-Even Pricing, widely used in all management sectors.*

1. Introdução

Com o cenário macroeconômico instável, o mercado de trabalho busca se reinventar e adicionar novas formas de renda, para que seja suprida a carência ocasionada pelo aumento dos preços de forma geral. No Brasil, a inflação sempre foi um problema à população, e no século XXI ela manteve-se constantemente entre os tópicos mais discutidos.

Na última década, houve a pandemia da Covid-19, uma doença respiratória aguda causada pelo vírus Sars-CoV-2, o qual se espalhava de forma aérea por meio das gotículas respiratórias que eram expelidas pela fala da pessoa infectada (OMS, 2023). Esta pandemia foi um terror na vida de muitas famílias devido a perda de entes queridos, crescimento de dívidas, fechamento de comércios etc.

Diante deste cenário, no qual os comércios permaneceram fechados para reduzir a chance de contágio populacional pelo vírus, diversas pessoas se viram na obrigação de empreender para que pudessem ter uma renda extra e/ou uma renda primária, nos casos de fechamento total do comércio do cidadão. A partir daí, surgiram as oportunidades de trabalhar remotamente em diversas empresas, para que pudessem continuar prestando seus serviços. Foi ali também que houve um aumento estrondoso na utilização de *Marketplaces* e houve a evolução do formato de serviço destes aplicativos. Entre tantos outros surgimentos e crescimentos de modelos de trabalho ou empreendimento, também surgiram muitos aspirantes a cozinheiros, que sustentavam esta paixão pela cozinha somente como um *hobby* e acabaram se encontrando na cozinha agora como uma forma de sustento.

Todavia, muitos destes cozinheiros se depararam com um grande empecilho na

hora de venderem suas receitas: a falta de conhecimento acerca da precificação delas. A precificação é um fator crucial e determinante para o sucesso dos empreendimentos em todos os âmbitos, logo, na culinária não seria diferente.

A democratização ao acesso de informações relacionadas às teorias de custeio de produtos são bastante nichadas e escassas para pessoas que não participam desta bolha, gerando uma precificação errônea por diversas vezes. Visto tal fato, o projeto apresentado neste artigo teve como objetivo desenvolver um aplicativo que busca trazer a democratização do acesso às informações sobre a teoria *cost-price*, bastante defendida por referências da área como: Eliseu Martins, Grady Booch e Steven Brag. De uma forma geral, o aplicativo visa reduzir as chances de erros na precificação das receitas feitas por pequenos restaurantes e empreendedores do ramo culinário, solicitando que o usuário insira as informações sobre sua receita, e o aplicativo trará qual o preço de venda do produto, já considerando o percentual de lucro definido pelo usuário.

O artigo em questão está estruturado em oito seções distintas. A Introdução fornece o contexto, o objetivo e a estruturação geral do texto. Em Gerenciamento de custos produtivos, há uma explanação básica sobre os gerenciamentos de custos empresariais, abordando as teorias de custo-preço (*Cost-Price*) e a forma de custeio adotada por microempreendedores. A seção sobre Desenvolvimento de aplicativos com *React Native* apresenta as tecnologias utilizadas na construção do aplicativo. Em *Firebase*, discute-se o banco de dados empregado no aplicativo. A seção de Engenharia de Requisitos explica os diagramas presentes no artigo. Prototipação descreve os métodos e tecnologias usados para criar o protótipo do aplicativo. Na seção de Resultados, são apresentados os resultados do projeto até o momento atual. Por fim, a Conclusão traz futuras melhorias a serem implementadas. Além disso, há uma seção de Referências ao final do artigo.

2. Gerenciamento de custos produtivos

Atualmente, a forma mais utilizada, empresarialmente, para controlar e gerir gastos e lucros, é a obtenção de um ERP (*Enterprise Resource Planning*) ou módulo ERP que seja voltado para controladoria e finanças. Algumas grandes empresas como TOTVS e SAP vendem estes produtos em uma grande quantidade, normalmente sendo adquiridos por empresas de médio e grande porte.

Um ERP é uma ferramenta poderosa para gestão geral, contemplando módulos aplicáveis em áreas como Manufatura, CRM (*Custom Relationship Management*), vendas, *Business Intelligence*, entre outras. Sua aplicação costuma abranger diversas regras de negócio bastante estruturadas e engessadas, a fim de fazer com que os produtos não apresentem falhas em sua construção (CONOVER, 2019).

Por outro lado, o ERP não é uma ferramenta conhecida por ter um bom custo-benefício, visto que seus módulos e a implantação destes pode ser caríssima. Além disso, as regras de negócio empresariais podem, nem sempre, ser um benefício a pequenas empresas, visto que seu engessamento pode provocar falhas e redundâncias em produtos mais simples.

Muitas vezes o ERP pode não satisfazer os gestores mais rústicos ou de empresas bastante pequenas, por conta da necessidade de ter cadastro de praticamente todas as compras da empresa, todas as ferramentas, todos os salários dos funcionários, horas trabalhadas, entre outras informações. No ramo gastronômico por exemplo, poderia ser

um stress aos chefes autônomos, visto que em grande parte das vezes, sua cozinha é a própria de sua casa. Nestes casos, um ERP para calcular os preços das receitas e estimar seus lucros seria algo totalmente inviável.

O gerenciamento de custos empresarial rotineiramente é pautado no modelo de precificação *cost-based pricing.* Este modelo tem várias formas de serem utilizadas, se adequando aos mais diversos cenários e necessidades. Dentre as formas, duas bastante comuns são:

a) *Cost-plus pricing*: Este modelo de precificação visa realizar o cálculo do custo de produção de um bem ou serviço e, ao fim, adicionar uma margem de lucro para que seja construído seu preço final. Este modelo é usado abrangentemente, por conta de sua fácil aplicabilidade. Para que haja um preço para seu bem ou produto, basta realizar a somatória de todos os custos (seja o custo de mão de obra, matéria prima, gasto energético, entre outros) e adicionar o percentual de lucro desejado para esta venda (ROBINSON; BERRY, 2023);
b) *Break-even pricing*: Usado mais comumente na égide orçamentária, este modelo calcula o preço das peças utilizando a demanda a ser produzida, considerando que, para que haja lucro, deverá ser suprida a necessidade do pagamento dos custos fixos e dos custos variáveis. O termo *Break-even* significa ponto de equilíbrio, o que dentro dos termos industriais indica um ponto no qual a receita obtida com as vendas está em equilíbrio com as despesas todas da empresa. Neste caso, quando a empresa se encontra em *Break-even*, não há lucro, pois, a somatória total das despesas está no mesmo patamar da somatória total das vendas. O lucro costuma ser estimado por peça, visando cobrir os custos de produção, os custos fixos da empresa e criar uma margem para a empresa, fazendo assim com que sua receita seja positiva e retorne lucro (BRAG, 2023).

Os custos fixos de produção são uma parcela razoavelmente grande dos custos totais da produção de um item. Dentro desta categoria, costumam serem abordados custos como o gasto com aluguel do estabelecimento, gasto com máquinas, energia elétrica, gasto com água e a depreciação de máquinas. Este custo tem diversos nomes no ramo empresarial, dentre eles: *Overhead Expense, Burden, Fixed Cost* e, mesmo todos sendo bastante semelhantes, existem sutis diferenças, como a adição do custo administrativo geral (*SG&A – Sales, General & Administrative Costs*) no caso do *Burden* (MARTINS, 2023).

De maneira geral, os custos fixos na gastronomia, são um pouco diferentes dos custos fixos empresariais. Comumente as empresas utilizam maquinários robustos e que demandam de um grande aporte energético, como injetoras e extrusoras. A utilização destas máquinas, acrescida ao gasto fixo com aluguel e amortizações, faz com que o *Overhead/Fixed Cost* empresarial seja uma razoável parcela do custo produtivo de um produto.

Em restaurantes e cozinhas *delivery*, o gasto com o *Fixed Cost* não é tão alto assim, visto que dia após dia os utensílios de cozinha e domésticos vêm recebendo aprimoramentos para a redução do consumo energético. Uma melhoria significativa nestes custos pode ser vista no caso dos gastos com geladeiras. Produtos produzidos no começo do século XXI, apontam um gasto cerca de 200% maior, segundo reportagem do site O Globo. Os dois itens mais utilizados em empreendimentos emergentes do ramo gastronômico são geladeira e fogão, os quais podem ser considerados como custos fixos

neste projeto (O GLOBO, 2016).

3. Desenvolvimento de Aplicativos com *React Native*

React Native foi a biblioteca escolhida para estruturar a construção do aplicativo *Price T'eat* por conta de sua grande desenvoltura para desenvolvimento mobile, gerado a partir de sua arquitetura exclusivamente direcionada para o desempenho em dispositivos móveis em multiplataformas.

3.1. JavaScript

JavaScript é uma linguagem de programação criada em 1995, por Brendan Eich. Inicialmente era utilizada pelo lado do cliente, para automatizar tarefas, gerir microsserviços *Front-End*, dinamizar telas WEB e criar requisições HTTP. A linguagem que já era bastante utilizada anteriormente, ganhou ainda mais notoriedade após a chegada do *Node.Js*, um interpretador JavaScript utilizado ao lado do servidor (MDN, 2024).

É uma linguagem de alto nível, dinâmica, interpretada e não tipada. Um ponto bastante positivo é sua versatilidade de paradigmas, podendo tanto ser Orientada a Objetos quanto Funcional. A sintaxe geral deriva da linguagem Java, e algumas funções e heranças são derivadas tanto de Scheme quanto de Self (FLANAGAN, 2020).

JavaScript recebeu seu nome inspirado em uma linguagem que estava em alta na época, chamada Java. Ambas as linguagens não têm correlação, somente uma breve semelhança sintática, entretanto, a ideia principal era alavancar a nova linguagem a partir do prefixo "Java". Abreviado como "JS", JavaScript tem uma alta versatilidade, permitindo que seja utilizada em diversos âmbitos programáveis, como WEB, *Mobile*, *Desktop* etc. (FLANAGAN, 2020).

3.2. Biblioteca React Native

Criado em 2011 por desenvolvedores da Meta (na época, Facebook) comandados por Jordan Walke, *React Native* é uma biblioteca *open source*, que tem como intuito principal facilitar a criação de aplicativos com alta fluidez e responsividade. Sua ideia geral é mesclar HTML com *JavaScript* (ou *TypeScript*) a fim de simplificar a criação dos aplicativos. Dentre os pontos positivos da utilização desta biblioteca, pode-se citar:

a) Flexibilidade na criação dos códigos, não limitando a usabilidade de nenhum componente;
b) Alta performance quando utilizada com DOM, reduzindo bastante o tempo de atualização das páginas;
c) Mobilidade, permitindo gerar tanto aplicativos quando aplicações WEB, com um grande reaproveitamento nos códigos.

Por JS (*JavaScript*) e TS (*TypeScript*) serem linguagens declarativas, *React Native* tem um grande foco em agilidade na atualização de seus componentes. Sua atualização é Hot Swap, visando acelerar o tempo de atualização da tela para que o desenvolvedor não perca rendimento (POLI, 2021).

Quando utilizada esta biblioteca, o processo de compilação do código é um pouco diferente do convencional. As etapas são:

a) Transpilação: utiliza Babel para converter qualquer versão do JavaScript para uma versão mais antiga, que possa ser lida pelo JavaScriptCore;
b) Empacotamento: o código transpilado é empacotado num arquivo APK

(Android) ou IPA (iOS).
c) Execução: é executado no dispositivo móvel, pelo Gradle (no Android) ou pelo Xcode (no iOS).

A compilação de códigos React Native pode ser otimizada a partir da utilização de ferramentas como o Metro Bundler ou o React Native CLI (PINHO; ESCUDELARIO, 2023).

4. Firebase

Plataforma de desenvolvimento de aplicativos que conta com um conjunto de serviços destinados a ajudar a criar, testar, lançar e escalar aplicativos Web e móveis. Entre os serviços disponibilizados, temos:

a) Banco de dados: *Cloud-firestore*, um banco de dados NoSQL;
b) Autenticação: *Firebase-Authentication*, parte da plataforma que é destinada a cuidar da autenticação dos usuários no software;
c) Notificações: *Cloud-messaging*, uma infraestrutura que permite o envio de mensagens Push para os usuários;

A plataforma foi criada pela Google com a finalidade de ser uma estrutura *back-end* completa para os seus usuários, recurso chamado de *BaaS (Backend as a Service*). Isso Facilita para que os desenvolvedores foquem exclusivamente na aplicação, tendo que atuar de maneira demasiada na infraestrutura de *back-end* (ALURA, 2023).

5. Engenharia de Requisitos

A engenharia de requisitos é pautada acerca da prototipação gráfica, diagramação do software e análise esmiuçada de detalhes técnicos. Está é uma parte importantíssima de um projeto que busca ter alta qualidade em seu produto pois é a partir dos requisitos levantados que podem ser definidas as capacidades e limitações presentes em um software (WIEGERS, 2022).

Tal engenharia pode ser dividida em 4 etapas, sendo (WIEGERS, 2022):

a) Levantamento de Requisitos: Aqui, são feitas entrevistas com usuários, clientes e *stakeholders*, para que o software possa atender integralmente ao que de fato precisa ser vendido. O *Stakeholder* é o responsável do produto pelo lado do cliente, podendo ser: um gerente da área que o software aborda, um gerente de TI, um desenvolvedor especialista da área etc. O levantamento de requisitos também pode ser feito a partir de visitas, onde uma pessoa responsável pelo software conviveria comercialmente ao lado da equipe e usuários do sistema durante alguns dias, a fim de encontrar falhas e rapidamente sanar estas lacunas;
b) Análise de Requisitos: Após feito o levantamento, agora deve haver a análise das informações e aqui elas serão esmiuçadas e validadas. As informações que forem relevantes ao software, terão andamento e passarão para a próxima etapa. As que forem simplesmente um luxo ou uma estimativa errônea, serão descartadas;
c) Especificação de Requisitos: Aqui, é criada a documentação dos requisitos que foram adiante após a fase de análise;
d) Verificação e Validação de Requisitos: Aqui é realizada a implementação dos requisitos especificados. São feitos novos testes para garantir que estão cumprindo com o que foi estimado e com o que foi especificado.

Dentro da engenharia de requisitos, existe uma linguagem padrão para que a diagramação do software seja algo universal. Isto é feito por meio da UML *(Unified Modeling Language),* uma linguagem de propósito geral, que permite a criação de modelagens para sistemas de qualquer nível de complexidade (BOOCH, 2023). Dentre seus diversos diagramas, os mais conhecidos são:

a) Diagrama de classes: Representa a estrutura estática do sistema, com seus objetos, atributos, operações e relacionamentos entre eles;
b) Diagrama de casos de uso: Descreve as funcionalidades do software do ponto de vista do usuário;
c) Diagrama de sequência: Mostra a dinâmica de comportamento do software e suas interações ao longo da atividade;
d) Diagrama de atividade: Representa o fluxo de controle do sistema, trazendo as atividades executadas e as relações entre elas;
e) Diagrama de componentes: Mostra a estrutura física do sistema, mostrando seus componentes e dependências entre eles.

Os diagramas presentes neste artigo foram feitos no Astah UML (BOOCH, 2023), ferramenta que permite a diagramação de softwares de forma gratuita para estudantes.

6. Prototipação

As prototipações presentes no artigo, foram realizadas a partir da ferramenta *web* Figma, voltada para a criação de protótipos de páginas, softwares, posts em redes sociais e demais modelagens e edições voltadas para a tecnologia. Sendo uma ferramenta muito forte para aprimoramento de ideias acerca de UI (*User Interface)* e UX (*User Experience),* conta com diversas facilidades par utilização, como as seguintes (FIGMA, 2023):

a) Ser arquitetado em *Cloud,* permitindo que o usuário possa editar em qualquer lugar, desde que tenha acesso à internet;
b) Ser uma ferramenta colaborativa, positiva para uso em projetos com múltiplos usuários;
c) Uma alta gama de recursos a serem utilizados pelo usuário.

No presente artigo, foi utilizada a versão gratuita do Figma, que supre com maestria as necessidades do projeto.

7. Resultados

Nesta seção serão expostos os diagramas UML feitos a partir dos requisitos do sistema, os protótipos de alta fidelidade, assim como uma breve explicação do fluxo de uso do programa. O nome escolhido para o aplicativo foi Price T'eat.

A seguir estão representados o Diagrama de Caso de Uso (Figura 1), o Diagrama de Atividades (Figura 2) e o Diagrama de Classes (Figura 3).

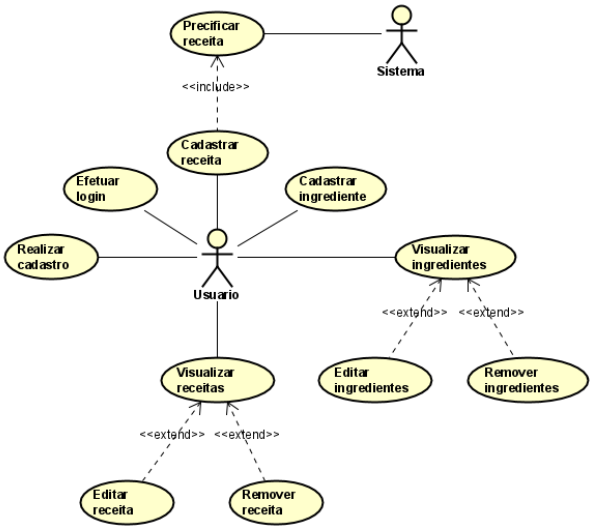


Figura 1: Diagrama de Caso de Uso do Software Price T'eat.

Fonte: Autoria própria.

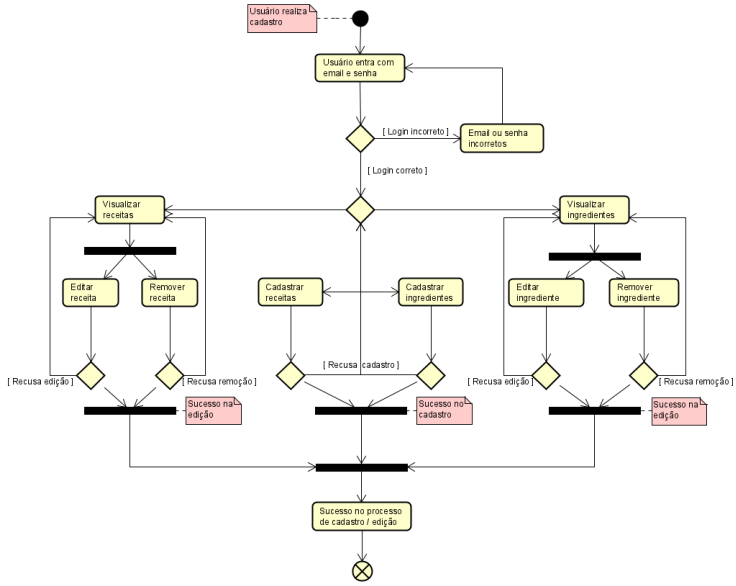


Figura 2: Diagrama de Atividades do Software Price T'eat.

Fonte: Autoria própria.

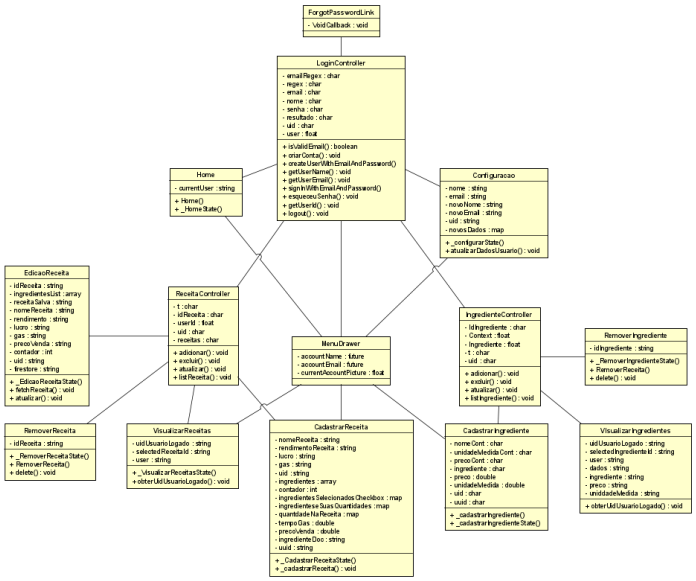
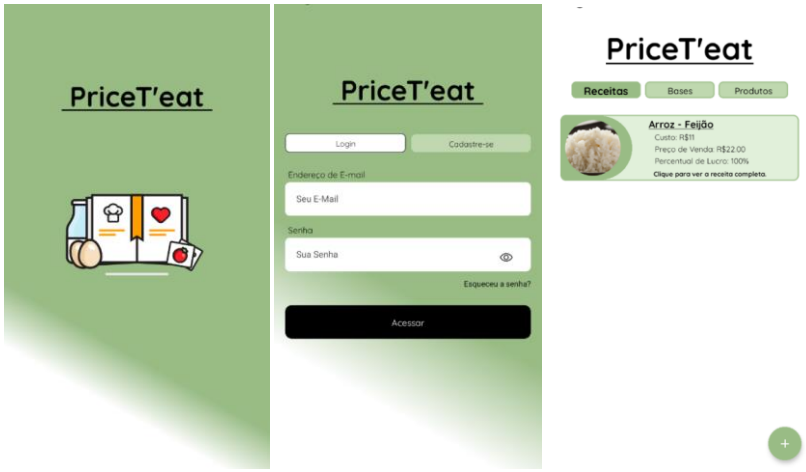


Figura 3: Diagrama de Classes do Software Price T'eat.

Fonte: Autoria própria.

Após realizar a modelagem funcional do sistema, foi criado seu protótipo e a seguir estão expostas das imagens iniciais do aplicativo. Inicialmente, nas Figuras 4a, 4b e 4c, estão mostradas, respectivamente, a tela de Login (conhecida como Splash Screen), a tela de cadastro de usuário e a tela de recuperação de senha. Já nas imagens 4d, 4e e 4f, há as imagens das telas que detalham tanto a base quanto a receita, e a tela de cadastro de receita, respectivamente. Nas imagens 4g, 4h e 4i, temos respectivamente a tela de cadastro de usuário, edição de produto e edição de base.



(a) **(b)** **(c)**

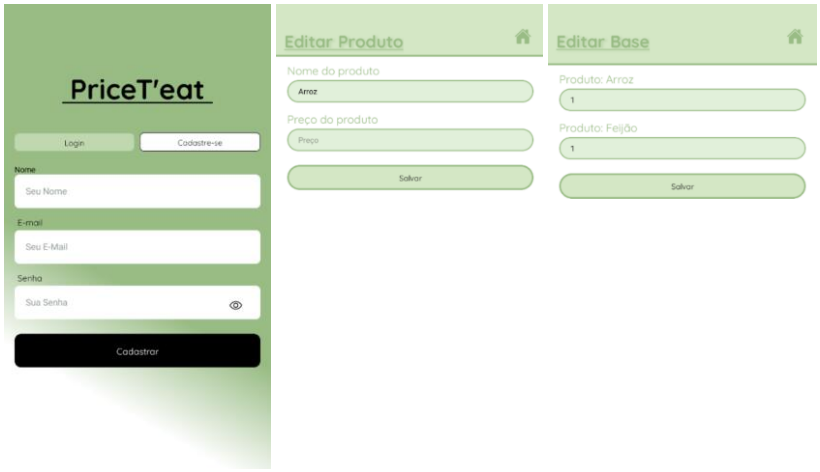
Figura 2: a) *Splash Screen;* b) Tela de *Login*; c) Tela de início.
Fonte: (Autoria própria, 2024)

Comentado [APZC1]: Espaçamento simples e acrescentar o ano

(d) **(e)** **(f)**

Figura 4: d) Detalhamento da base; e) Detalhamento da Receita; f) Cadastro de Receita.
Fonte: (Autoria própria, 2024)

Comentado [AC2]: Espaçamento simples e acrescentar o ano



(g) **(h)** **(i)**

Figura 4: g) Cadastro de usuário; h) Edição de produto; i) Edição de base.
Fonte: (Autoria própria, 2024)

Comentado [AC3]: Espaçamento simples e acrescentar o ano

8. Conclusão

O aplicativo *Price T'eat* será uma boa forma de reduzir o gargalo nos lucros de pequenos restaurantes e empreendedores do ramo culinário. Sendo um facilitador para a construção dos preços de venda dos produtos, o usuário não precisará se preocupar com a parte dos cálculos, somente deverá cadastrar corretamente as informações solicitadas pelo aplicativo:

a) Nome do material;
b) Quantidade usada na receita;
c) Preço pago no material;
d) Unidade de medida do material;
e) Percentual de lucro desejado na venda.

Visto que todas as informações podem ser adquiridas no verso da embalagem ou no cupom fiscal da compra, o uso do *Price T'eat* é democrático e busca abranger públicos de diversas classes. Todavia, a partir dos resultados obtidos neste estudo, há-se a sugestão para a realização de trabalhos complementares no futuro, como sendo:

a) A finalização do desenvolvimento deste aplicativo, visto que atualmente encontra-se com uma versão *beta* sem todas as suas finalidades;
b) A utilização de CI/CD na implementação de melhorias dentro do aplicativo, buscando uma melhor experiência e usabilidade ao público;
c) A adição de um diagrama de casos de testes, analisando quais os tipos de dados que são aceitos nos campos *inputs* do aplicativo;
d) A atualização automática na *Flatlist* de receitas, visto que este componente não

trabalha bem com navegações nativas, as quais estão sendo utilizadas neste projeto;

Referências

ALURA, O que é Firebase? Para que serve, principais características e um Guia dessa ferramenta Google, 2023. Disponível em <https://www.alura.com.br/artigos/firebase>. Acesso em 16 de outubro de 2023.

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar (2005). The Unified Modeling Language: A Reference Manual. 3. ed. Addison-Wesley Professional.

BRAG, Steven (2023). Manual de políticas e procedimentos contábeis: um plano para executar um departamento eficaz e eficiente. 4. ed. São Paulo: Atlas.

CONOVER, John W (2019). ERP: Enterprise Resource Planning. 2. ed. Upper Saddle River, NJ: Prentice Hall.

FIGMA, Documentação do Figma (2023). Disponível em: <https://figma.com/hc/en-us>. Acesso em: 30 de outubro de 2023.

FIREBASE, Documentação do Firebase (2023). Disponível em: <https://firebase.google.com/docs/>. Acesso em: 16 de outubro de 2023.

FLANAGAN, David. (2020). JavaScript: O Guia Definitivo. 7ª ed. Rio de Janeiro: Alta Books.

IBGE – INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA. Índice Nacional de Preços ao Consumidor Amplo (IPCA). Disponível em: <https://www.ibge.gov.br/estatisticas/economicas/precos-e-custos/9256-indice-nacional-de-precos-ao-consumidor-amplo.html>. Acesso em: 25 de agosto de 2023.

MARTINS, Eliseu (2023). Contabilidade de custos. 18. ed. São Paulo: Atlas.

MDN Web Docs. “Estrutura de Documento e Sites.” MDN Web Docs.

< https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/First_steps/What_is_JavaScript#defini%C3%A7%C3%A3o_de_alto_n%C3%ADvel>. Acesso em: 19 de março de 2024.

NAGLE, Thomas T.; HOLDEN, Reed K (2023). Pricing for profit. 6. ed. New York: McGraw-Hill Education.

O GLOBO. Trocar aparelhos antigos compensa para gastar menos energia. Disponível em: <https://oglobo.globo.com/economia/defesa-do-consumidor/trocar-aparelhos-antigos-compensa-para-gastar-menos-energia-19139138>. Acesso em: 25 de agosto de 2023.

Organização Mundial da Saúde (OMS). COVID-19. Disponível em: < https://www.who.int/emergencies/diseases/novel-coronavirus-2019/question-and-answers-hub/q-a-detail/coronavirus-disease-covid-19 >. Acesso em: 17 de outubro de 2023.

PINHO, Diego Martins de; ESCUDELARIO, Bruna. React Native: Guia Completo de Desenvolvimento Mobile. 1. ed. São Paulo: Casa do Código, 2023.

POLI Júnior (POLI). React native: saiba mais sobre essa biblioteca JavaScript.

Disponível em: <https://polijunior.com.br/blog/react-native-aprenda-sobre-biblioteca-javascript/#:~:text=O%20React%20Native%20%C3%A9%20uma,engenheiro%20de%20software%20Jordan%20Walke.>. Acesso em: 25 de Março de 2024.

WIEGERS, Karl E.; BACH, Joy L (2022). Engenharia de Requisitos: Uma Abordagem Prática. Elsevier.