DESENVOLVIMENTO DE APLICATIVO PARA CONTROLE FINANCEIRO COM MÚLTIPLAS CONTAS BANCÁRIAS

Leonardo Lindroth[1]; Romulo Castillo[2]

[1]*Universidade Federal de Santa Catarina, Joinville, Brasil (leolindroth14@gmail.com)*

[2]*Universidade Federal de Santa Catarina, Joinville, Brasil (romulo.castillo@ufsc.br)*

Resumo: A digitalização bancária e a expansão dos bancos digitais ampliaram a oferta de serviços financeiros, mas tornaram mais complexo o controle de múltiplas contas por um mesmo usuário. Este trabalho apresenta o desenvolvimento de um aplicativo mobile capaz de centralizar contas e transações em um único ambiente. Adotou-se desenvolvimento ágil de um produto mínimo viável, com elicitação de requisitos, modelagem UML e uso de React Native e Firebase. O resultado é uma aplicação funcional, com 85,70% dos requisitos atendidos.

Palavras-chave: Controle Financeiro; Desenvolvimento Mobile; Engenharia de Software; React Native; Produto Mínimo Viável.

INTRODUÇÃO

O controle financeiro sempre desempenhou papel essencial na vida das pessoas, servindo como ferramenta fundamental para o equilíbrio entre ganhos e gastos e para a tomada de decisões econômicas conscientes. Com o avanço da tecnologia e a digitalização dos serviços bancários, especialmente com a popularização dos bancos digitais, o gerenciamento financeiro passou a envolver uma quantidade crescente de contas, aplicações e transações distribuídas entre diferentes instituições. Essa multiplicidade, embora traga vantagens em termos de competitividade e acesso a serviços personalizados, gera complexidade e falta de centralização, dificultando a visualização global das finanças pessoais e empresariais.

Nesse cenário, evidencia-se a necessidade de ferramentas tecnológicas capazes de integrar e consolidar informações financeiras de diferentes instituições em um único ambiente digital, oferecendo praticidade, segurança e clareza no controle das movimentações. O presente trabalho tem como objetivo o desenvolvimento de um software de controle financeiro com múltiplas contas bancárias, estruturado como aplicativo mobile. O sistema foi concebido para centralizar contas e operações bancárias, permitindo que o usuário acompanhe saldos, realize transferências, registre pagamentos e controle seus ganhos e gastos.

Para a construção do sistema, foram utilizadas tecnologias amplamente adotadas na indústria, como *Node.js* para o *backend*, *React Native* para o desenvolvimento mobile e *Firebase* como solução de banco de dados e autenticação. O projeto seguiu os princípios do desenvolvimento ágil, com a criação de um Produto Mínimo Viável (MVP), que permite testar e validar as funcionalidades essenciais antes de futuras expansões. A metodologia adotou boas práticas de engenharia de software, com levantamento de requisitos, elaboração de diagramas UML e prototipação, resultando em um aplicativo que une design moderno, usabilidade e confiabilidade técnica.

FUNDAMENTAÇÃO TEÓRICA

Importância do controle financeiro

O controle financeiro assenta-se sobre um pilar central: administrar os ganhos de modo que não sejam superados pelos gastos, formando um sistema que se equilibra ou gera margem de lucro. Contudo, tais

informações nem sempre alcançam toda a população, uma vez que buscar orientação sobre gestão de finanças não integra o cotidiano da maioria das pessoas, agravado pela ausência de uma cultura coletiva organizada em torno do tema (Banco Central do Brasil, 2013). Essas lacunas informacionais e sociais produzem uma falsa sensação de domínio sobre a gestão financeira pessoal e familiar.

Os dados confirmam a fragilidade do cenário. Entre as famílias brasileiras, pesquisas indicam que três em cada quatro enfrentam alguma dificuldade para chegar ao fim do mês com seus rendimentos (Banco Central do Brasil, 2013). No segmento empresarial, o Sebrae (2023) aponta que os microempreendedores individuais registram a maior taxa de mortalidade entre os pequenos negócios, com 29% encerrando as atividades após cinco anos, seguidos pelas microempresas, com 21,6%, e pelas empresas de pequeno porte, com 17%.

Verifica-se, portanto, que cerca de 75% das pessoas físicas sentem insegurança quanto às suas rendas e que a mortalidade de pequenos negócios é elevada. Embora múltiplos fatores influenciem esses indicadores, sendo difícil isolar a contribuição da carência de conhecimento em gestão financeira, o controle financeiro desponta como ação ao alcance do indivíduo, independentemente de influências externas, ganhando força como projeto pessoal e empresarial.

Sistema bancário e digitalização

Na era digital, a transformação digital consiste em modernizar processos internos por meio das tecnologias disponíveis, buscando eficiência e redução de custos. Trata-se de tendência mundial, e os bancos perseguem a digitalização bancária, que converte os serviços para o formato on-line e digital, bem como reestrutura os processos internos que sustentam essa mudança (Banco Central do Brasil, 2020). É com o advento das aplicações web e, sobretudo, mobile que a facilitação bancária se consolida, permitindo que uma pessoa abra conta e acesse funcionalidades sem sair de casa.

O uso de meios digitais para transações financeiras no Brasil cresceu de forma acelerada na última década, impulsionado pelo arcabouço legal que fomentou os mercados de arranjos e instituições de pagamento, pelas inovações tecnológicas, pela introdução do Pix e pelas medidas de isolamento social adotadas durante a pandemia de Covid-19 (Banco Central do Brasil, 2022). Com a crescente oferta de instituições de pagamento e bancos digitais, os usuários passaram a contratar serviços de múltiplas instituições, formando uma carteira de contas a administrar. O volume dificulta o controle financeiro, pois demanda mais tempo de análise; assim, centralizar as informações em um único local reduz esse tempo e facilita a gestão.

Entretanto, a segurança digital é o tópico mais sensível em um sistema centralizado de contas. A integração entre bancos por meio de cadastro único não é trivial, dadas as regras do Banco Central do Brasil aplicáveis às instituições de pagamento e o alinhamento necessário com a Lei Geral de Proteção de Dados (LGPD), que impõe penalidades em caso de vazamento. Nesse contexto, o Banco Central anunciou, em 2021, a iniciativa Open Finance, estruturada em quatro fases graduais: disponibilização de informações sobre canais, produtos e serviços; compartilhamento de dados do consumidor mediante consentimento revogável; acesso a serviços financeiros como pagamentos e

propostas de crédito sem recorrer aos canais das instituições; e inclusão de novos dados e produtos, como câmbio, investimentos, seguros e previdência (Open Finance, 2025). Atualmente na Fase 4, plenamente operacional, a tecnologia permite centralizar dados mediante autorização do cliente, sincronizando informações bancárias e ampliando a liberdade de uso das contas.

Sistema de inspiração

Realizou-se pesquisa sobre a posição do mercado quanto a esse tipo de gerenciamento, a fim de identificar competidores consolidados. Destaca-se o *Organizze*, fundado em 2009 com a proposta de oferecer gerenciamento financeiro em um único local. A plataforma, disponível nas versões web e mobile, foi utilizada neste trabalho como inspiração e base para o reconhecimento das funcionalidades esperadas em um software de cunho financeiro. Seu modelo de rentabilização baseia-se em planos com funcionalidades distintas, sendo que os planos superiores empregam o Open Finance para integrar-se aos bancos parceiros e automatizar controles que, de outra forma, seriam manuais. A interface mobile mostrou-se intuitiva, com cadastro rápido e adição de contas e cartões já na tela inicial, o que orientou as decisões de projeto do presente sistema.

MATERIAL E MÉTODOS

Optou-se por uma abordagem clássica e segura na etapa de desenvolvimento, com foco na elicitação de requisitos, atividade relacionada à descoberta e ao entendimento das necessidades de um sistema. Conforme Valente (2022):

> Diversas técnicas podem ser usadas para elicitação de requisitos, incluindo entrevistas com stakeholders, aplicação de questionários, leitura de documentos e formulários da organização que está contratando o sistema, realização de workshops com os usuários, implementação de protótipos e análise de cenários de uso.

Adotou-se o modelo de produto mínimo viável (MVP), definido por Ries (2011) como a versão do produto que permite uma volta completa do ciclo construir-medir-aprender, com o mínimo de esforço e o menor tempo de desenvolvimento, ainda que careça de recursos que se mostrem necessários mais tarde. Para a elicitação, procedeu-se à análise do software Organizze, buscando compreender como a interface se traduz em funcionalidades e arquitetura.

Após a elicitação, empregou-se linguagem técnica para iniciar a prototipação. A Linguagem de Modelagem Unificada (UML) provê a arquitetos e engenheiros de software ferramentas para analisar, idealizar e implementar sistemas, integrando modelos de negócio e processos similares (OMG, 2017). Por meio dela, os requisitos levantados traduzem-se em modelos interpretáveis e conversíveis em código.

Para atingir o MVP em prazo curto, empregou-se desenvolvimento ágil orientado à construção de um protótipo estruturado, apoiado em um *framework* de aplicação orientado a objetos, caracterizado por modularidade, reusabilidade, extensibilidade e inversão de controle (Fayad; Schmidt, 1997). Utilizou-se o *Node.js*, software de código aberto e multiplataforma baseado no interpretador V8, que permite executar JavaScript fora do navegador e dispõe de gerenciador de pacotes que simplifica a gestão de dependências, tornando a construção de sistemas mais rápida, organizada e auditável.

Sobre essa base, adotou-se o *React Native*, definido por Eisenman (2017) como framework em JavaScript que permite construir aplicações mobile com renderização nativa para iOS e Android, baseado no React, mas voltado às plataformas móveis, convertendo JavaScript e XML em Objective-C e Java. Aplicou-se ainda o *Expo*, que simplifica o desenvolvimento ao fornecer roteamento por estrutura de arquivos, biblioteca padrão de módulos nativos e o serviço *Expo Application Services* (EAS), responsável pela infraestrutura de publicação nas lojas de aplicativos.

Para a persistência de dados, utilizou-se modelo não relacional (NoSQL), que opera com estruturas distintas do formato linha-coluna e permite escalabilidade horizontal. Ainda assim, para idealizar a estrutura inicial na fase de concepção, adotou-se o modelo entidade-relacionamento, mais comum em bancos relacionais, empregando os conceitos de chave-valor e documento. A infraestrutura é mantida pelo *Firebase*, que fornece o Cloud Firestore para controle dos dados e o Authentication para autenticação, ambos com uso gratuito até determinada escala. Pelo módulo Authentication realiza-se o cadastro e o login com e-mail e senha, mantendo um token de sessão que evita reautenticação a cada abertura do aplicativo.

DESENVOLVIMENTO

A etapa de pré-desenvolvimento concentra-se em tangibilizar o planejamento, atrelando-o às definições iniciais de requisitos, que constituem a ponte entre um problema do mundo real e o sistema de software que o soluciona (Valente, 2020). O planejamento é fundamental para minimizar problemas, visto que quanto mais tarde o defeito aparece, mais custosa é sua correção. Segundo Valente (2020), de nada adianta um sistema com o melhor design, implementado na linguagem mais moderna e com alta cobertura de testes, se ele não atender às necessidades de seus usuários.

Requisitos

Os requisitos funcionais descrevem o que o sistema deve fazer, dependendo do tipo de software, dos usuários a que se destina e da abordagem da organização ao redigi-los (Sommerville, 2007). O Quadro 1 sintetiza os requisitos funcionais elicitados.

Quadro 1 – Requisitos funcionais

Req.	**Descrição**
RF01	Permitir a criação de conta de usuário
RF02	Permitir autenticação
RF03	Possibilitar a adição de contas bancárias
RF04	Permitir a realização de transferência bancária
RF05	Permitir o pagamento de conta
RF06	Permitir a inserção de recebimentos
RF07	Ter visualização centralizada das operações
RF08	Alertar com reforço visual uma ação realizada

Fonte: o autor (2025).

As contas bancárias constituem as entidades centrais do sistema e podem ser de titularidade do próprio cliente ou de terceiros. A partir delas, o sistema controla três tipos de transação: transferência, pagamento e recebimento. A transferência consiste na troca de valores entre contas, podendo ocorrer por TED, DOC ou PIX; as duas primeiras estão sempre disponíveis, pois os dados bancários são obrigatórios no cadastro, ao passo que o PIX permanece desabilitado quando não há chave associada. Os pagamentos permitem quitar boletos ou chaves PIX, sendo registrados como gastos, enquanto os recebimentos registram os ganhos, em caráter antagônico aos anteriores. Essas transações viabilizam o monitoramento de gastos e ganhos ao longo do tempo, centralizando as informações em um único local.

Dois requisitos inicialmente previstos foram descartados na elicitação: a integração com o Open Finance, que exigiria elevado nível de segurança e reconhecimento pelo Banco Central como instituição de pagamento, inviável nesta versão; e a notificação de ações em segundo plano, dispensável uma vez que todos os informes passaram a ser manuais.

Os requisitos não funcionais, por sua vez, não se relacionam diretamente às funções específicas do sistema, podendo referir-se a propriedades emergentes, como confiabilidade e tempo de resposta, ou a restrições de dispositivos e representações de dados (Sommerville, 2007). O Quadro 2 apresenta os requisitos não funcionais, que delimitam as tecnologias empregadas e as condições de disponibilidade e responsividade.

Quadro 2 – Requisitos não funcionais

Req.	**Descrição**
RNF01	Framework React Native
RNF02	Banco de dados Firebase
RNF03	Aplicativo disponível na Play Store
RNF04	Autenticação via Firebase
RNF05	Disponibilidade a partir do Android 5.0
RNF06	Responsividade para web

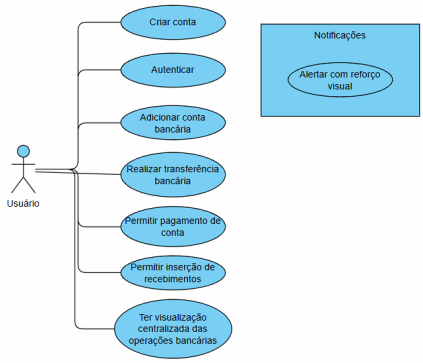
Fonte: o autor (2025).

Complementarmente, as regras de negócio não constituem necessidades inerentes do cliente, mas condicionam a execução das funcionalidades. Entre elas, destacam-se a criação de conta bancária para controle manual quando não há integração com o Open Finance; o bloqueio de transações com valores superiores ao montante da conta de origem ou ao limite da instituição; o impedimento de pagamento por PIX quando não há chave cadastrada na conta de destino; e a exibição de saldo restrita às contas integradas ou de controle interno.

Diagramas de caso de uso e de atividades

O diagrama de casos de uso apresenta uma visão externa e geral das funcionalidades que o sistema oferece, sem detalhar sua implementação, auxiliando na identificação e compreensão dos requisitos e na documentação das características e serviços desejados pelo usuário (Guedes, 2011). A Figura 1 traduz os requisitos levantados em casos de uso, evidenciando sete funcionalidades de execução do usuário e uma de execução do subsistema de notificações, de controle interno.

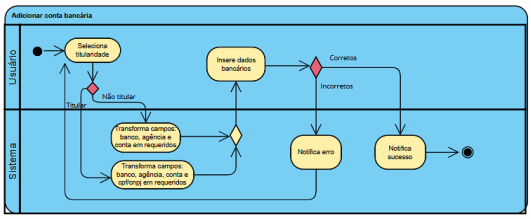
Figura 1 – Diagrama de caso de uso



Fonte: o autor (2025).

Para compreender como um caso de uso deve ser implementado, recorre-se ao diagrama de atividades, que descreve os passos a serem percorridos para a conclusão de uma atividade específica, concentrando-se na representação do fluxo de controle (Guedes, 2011). Dada a semelhança entre os fluxos, modelou-se o caso de uso mais complexo, a adição de conta bancária (Figura 2). Nele, o primeiro nó de decisão refere-se à titularidade da conta, escolha que determina quais dados serão obrigatórios no nó seguinte; ao final, validados os dados, emite-se notificação de sucesso e o fluxo se encerra.

Figura 2 – Diagrama de atividades: adicionar conta bancária

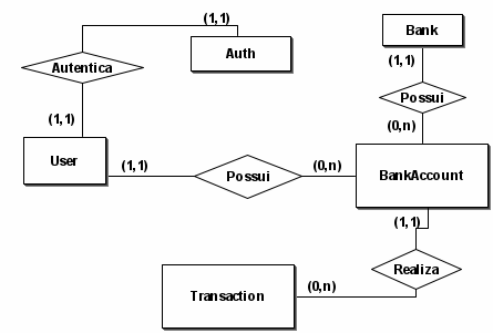


Fonte: o autor (2025).

Modelo entidade-relacionamento

Os diagramas UML fornecem as informações necessárias para o emprego do modelo entidade-relacionamento, ferramenta relevante no desenvolvimento de sistemas complexos e modelo conceitual utilizado na Engenharia de Software para descrever as entidades de um domínio de negócios, seus atributos e relações (Franck; Pereira; Dantas, 2021). Embora o React Native siga as diretrizes do JavaScript e utilize objetos, sua orientação a eventos torna dispensável o diagrama de classes. Elaboraram-se, assim, os diagramas conceitual (Figura 3) e lógico (Figura 4), que descrevem a estrutura de armazenamento e definem as tabelas e propriedades implementadas no modelo físico do Firebase (Figura 5).

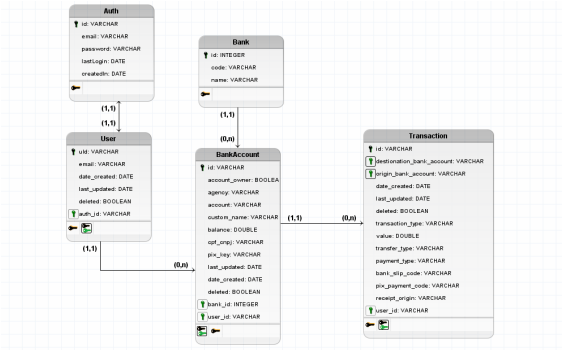
Figura 3 – Diagrama entidade-relacionamento conceitual



Fonte: o autor (2025).

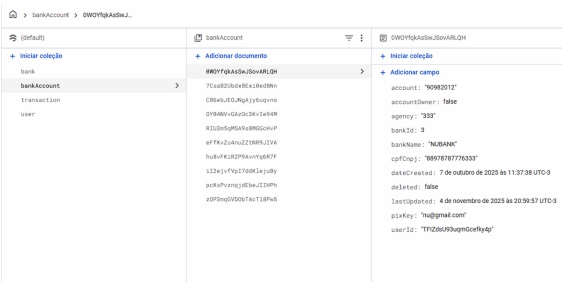
Esse tipo de modelo é mais aplicado a bancos SQL, relacionais e escalonados verticalmente por chaves primárias e estrangeiras. No paradigma NoSQL adotado, ele também explica as relações entre entidades; contudo, como as tabelas escalonam horizontalmente e admitem tipos não primitivos, seria possível agrupá-las para reduzir o tempo de busca. Optou-se por manter o modelo entidade-relacionamento por sua clareza, evitando a técnica de agrupamento permitida pelo conceito NoSQL.

Figura 4 – Diagrama entidade-relacionamento lógico



Fonte: o autor (2025).

Figura 5 – Modelo físico no Firebase



Fonte: o autor (2025).

Design de telas

A partir das definições de modelo físico e do mapeamento de casos de uso, elaboraram-se as telas principais, estabelecendo o escopo de acesso às funcionalidades. Os componentes foram concebidos em escalas unitárias, de modo a serem reaproveitados, simplificando a manutenção e garantindo coesão visual, com vistas à construção de um *Design System*. O fluxo completo, criado em software de layouts, corresponde ao ciclo do objeto "conta bancária", abrangendo criação, visualização, edição e exclusão, e serviu de diretriz para os demais casos de uso.

RESULTADOS E DISCUSSÃO

A partir das definições e conceitos estabelecidos na fase de desenvolvimento, o sistema foi construído e aprimorado ao longo do processo, tomando como ponto de partida os diagramas, os requisitos e o design de telas inicial.

Sistema desenvolvido

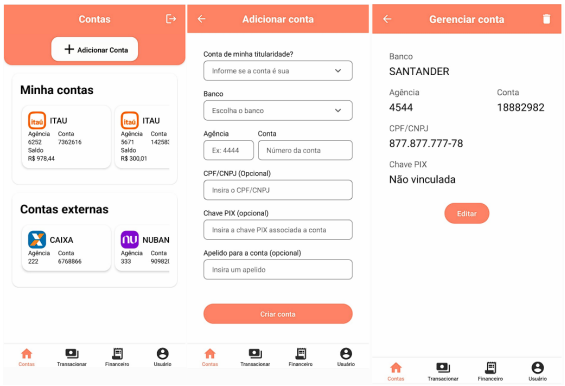
O sistema organiza-se em cinco telas principais: uma de cadastro e login em conjunto e quatro telas autenticadas que compõem o gerenciamento das informações. Por tratar-se de um MVP, a estrutura de cadastro foi mantida simples, exigindo apenas e-mail e senha (Figura 6).

Figura 6 – Tela de login e cadastro

Fonte: o autor (2025).

Realizada a autenticação, o usuário é direcionado à tela inicial, que dispõe das abas de navegação no rodapé, a indicação textual da tela corrente e o botão de saída no canto superior direito. No primeiro acesso, apenas o botão de adicionar conta é exibido, conduzindo à criação das entidades de conta bancária que servirão de base às transações. O formulário de criação contempla os bancos cadastrados e os campos necessários, admitindo contas de titularidade do cliente ou de terceiros. Após a criação, a tela inicial exibe todas as contas cadastradas e permite consultar, editar ou excluir cada uma delas (Figura 7).

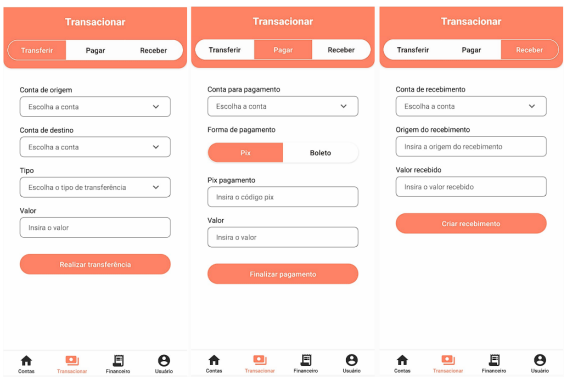
Figura 7 – Telas de contas bancárias



Fonte: o autor (2025).

O segundo menu concentra as transações, realizadas por três tipos: transferência, entre contas do cliente ou de terceiros previamente cadastradas; pagamento, referente a boletos ou chaves PIX, registrado como saída; e recebimento, entrada manual que informa ganhos ou rendimentos. Para cada tipo há campos distintos, e um seletor no cabeçalho ajusta o formulário conforme a necessidade de registro (Figura 8).

Figura 8 – Telas de transações

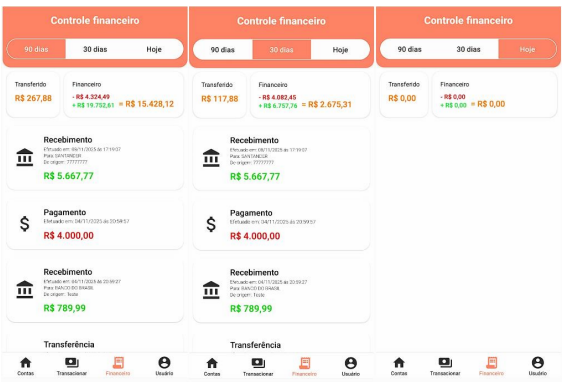


Fonte: o autor (2025).

As transações constituem as entidades mais relevantes para a gestão financeira, pois é por meio delas que se apuram ganhos e perdas ao longo do tempo. No menu financeiro, é possível visualizar as transações em períodos específicos — hoje, 30 dias e 90 dias —, contados a partir da data atual. Selecionado o período, dois elementos visuais consolidam os dados acima da listagem, informando o montante transferido e a quantidade de pagamentos e

recebimentos, o que permite identificar rapidamente se houve ganho ou perda no intervalo (Figura 9). Há ainda uma tela dedicada à conta de usuário, que exibe os dados cadastrais e possibilita o encerramento da sessão.

Figura 9 – Telas de controle financeiro



Fonte: o autor (2025).

Requisitos atendidos

O Quadro 3 consolida o atendimento dos requisitos funcionais e não funcionais.

Quadro 3 – Resultado dos requisitos

Req.	**Resultado**
RF01 a RF08	Atendidos
RNF01 – React Native	Atendido
RNF02 – Firebase	Atendido
RNF03 – Play Store	Não atendido
RNF04 – Autenticação	Atendido
RNF05 – Android 5.0	Atendido
RNF06 – Responsividade web	Parcialmente atendido

Fonte: o autor (2025).

Todos os oito requisitos funcionais foram atendidos, proporcionando experiência completa ao usuário mediante registro manual de suas finanças em um período de controle definido. Quanto aos requisitos não funcionais, os quatro relativos a frameworks e sistemas foram cumpridos, satisfazendo as definições de segurança e disponibilidade planejadas.

Dois requisitos, contudo, não foram plenamente atendidos. A responsividade para web foi apenas parcialmente implementada, com correspondência em alguns elementos, mas sem aprimoramento de layout e design, o que dificulta o uso nesse formato. A publicação na Play Store não se concretizou, uma vez que o modelo de distribuição do Google exige uma série de testes e aprovações que inviabilizaram o processo no prazo do projeto; o aplicativo permanece disponível para instalação direta por meio do arquivo instalável presente no repositório do projeto.

Agrupando os resultados, alcançou-se 85,70% de requisitos atendidos, com 7,15% parcialmente atendidos e 7,15% não atendidos. A integração com o Open Finance, não implementada nesta versão, permanece como principal oportunidade de aprimoramento, capaz de agregar valor ao produto e elevá-lo a patamar competitivo de mercado.

CONCLUSÃO

O desenvolvimento do software de controle financeiro com múltiplas contas bancárias demonstrou a viabilidade técnica e prática de integrar tecnologias modernas em um sistema acessível e funcional. A aplicação atendeu à maior parte dos requisitos propostos, oferecendo experiência fluida de gerenciamento por meio de cadastro, autenticação, criação de contas, transações e visualização centralizada das informações.

Embora o projeto ainda comporte aprimoramentos, como notificações automáticas e maior responsividade web, os resultados obtidos comprovam o potencial da solução como ferramenta útil para o controle financeiro pessoal e empresarial. O trabalho contribui, assim, com uma proposta tecnológica aplicável, orientada à redução da insegurança financeira e da mortalidade de pequenos negócios. Pesquisas futuras poderão explorar integrações mais profundas com o Open Finance e expandir

as funcionalidades para oferecer análises preditivas e relatórios personalizados, dispensando o registro totalmente manual e ampliando o impacto da solução.

DECLARAÇÃO DE USO DE INTELIGÊNCIA ARTIFICIAL

Durante a elaboração deste trabalho utilizou-se o assistente Claude (Anthropic) como ferramenta de apoio, restrita à correção gramatical e ortográfica, ao resumo de trechos previamente redigidos pelo autor e à adequação do texto ao formato editorial exigido. A concepção do projeto, o levantamento de requisitos, a modelagem do sistema, o desenvolvimento da aplicação, a execução dos testes, a análise dos resultados e as conclusões são de autoria integral do autor.

REFERÊNCIAS

BANCO CENTRAL DO BRASIL. Caderno de Educação Financeira: gestão de finanças pessoais. Brasília: BCB, 2013.

BANCO CENTRAL DO BRASIL. Fintechs de crédito e bancos digitais. Brasília: BCB, 2020.

BANCO CENTRAL DO BRASIL. Relatório de Economia Bancária: evolução de meios digitais para realização de transações de pagamento no Brasil. Brasília: BCB, 2022.

EISENMAN, B. Learning React Native. 2. ed. Sebastopol: O'Reilly Media, 2017.

FAYAD, M. E.; SCHMIDT, D. C. Object-oriented application frameworks. Communications of the ACM, v. 40, n. 10, p. 32-38, 1997.

FRANCK, K. M.; PEREIRA, R. F.; DANTAS, J. V. Diagrama entidade-relacionamento: uma ferramenta para modelagem de dados conceituais em Engenharia de Software. Research, Society and Development, v. 10, n. 8, 2021.

GUEDES, G. T. A. UML 2: uma abordagem prática. 2. ed. São Paulo: NovaTec, 2011.

OBJECT MANAGEMENT GROUP. Unified Modeling Language Specification, versão 2.5.1. Needham: OMG, 2017.

OPEN FINANCE. Relatório anual 2024. Brasília: Open Finance, 2025.

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

SEBRAE. Sobrevivência das empresas no Brasil. Brasília: Sebrae, 2023.

SOMMERVILLE, I. Engenharia de software. 8. ed. São Paulo: Pearson, 2007.

VALENTE, M. T. Engenharia de software moderna: princípios e práticas para desenvolvimento de software com produtividade. Belo Horizonte: Edição Independente, 2020.