PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS

ESCOLA POLITÉCNICA

ESTUDO DE FERRAMENTAS PARA O DESENVOLVIMENTO DE UM APLICATIVO MOBILE PARA BANCO DIGITAL

Goiânia

DEZEMBRO/2021

REVERSON THAIAN BATISTA DOS SANTOS

ESTUDO DE FERRAMENTAS PARA O DESENVOLVIMENTO DE UM APLICATIVO MOBILE PARA BANCO DIGITAL

Trabalho de Conclusão de Curso apresentado à Escola Politécnica, como requisito para a obtenção do grau de Cientista da Computação no Curso de Ciência da Computação, na Pontifícia Universidade Católica de Goiás.

Orientadora: Prof.ª Msc Lucília Gomes Ribeiro

Goiânia

2021

REVERSON THAIAM BATISTA DOS SANTOS

ESTUDO DE FERRAMENTAS PARA O DESENVOLVIMENTO DE UM APLICATIVO MOBILE PARA BANCO DIGITAL

Este Trabalho de Conclusão de Curso julgado adequado para obtenção o título de Bacharel em Ciência da Computação, e aprovado em sua forma final pela Escola Politécnica, da Pontifica Universidade Católica de Goiás, em 01/12/2021.

Prof. Me. Ludmilla Reis Pinheiro dos Santos
Coordenador(a) de Trabalho de Conclusão de Curso

Banca examinadora:

Orientador(a): Prof. Me. Lucília Gomes Ribeiro

Prof. Me. Anibal Santos Jukemura

Prof. Me. Fernando Gonçalves Abadia

Goiânia
2021

RESUMO

Com o aumento da população com acesso à internet, principalmente utilizando dispositivos móveis, houve a necessidade do mercado bancário se atualizar diante da tecnologia para oferecer melhores serviços. Então foi se criando os bancos digitais oferecendo produtos e serviços cada vez melhores, diminuindo o monopólio dos grandes bancos tradicionais e aumentando a concorrência. O desenvolvimento de um aplicativo bancário se mostrava impraticável para ser construído, em tempo hábil e equipe reduzida. Atualmente as linguagens nos proporcionam o mesmo recurso em tempo reduzido e menor custo de equipe. Portanto, este trabalho é um projeto de software que simula um aplicativo de um banco digital, onde o foco principal é o controle da conta para fazer operações tradicionais, como transferências, pagamentos, histórico, depósitos, levando em consideração toda estrutura de segurança para fornecer um serviço confiável. Este trabalho apresenta os requisitos, a implementação e a documentação do aplicativo de um banco digital, bem como as tecnologias utilizadas. Para a implementação foram utilizados o framework ReactJs e para a construção das interfaces, a linguagem de programação Javascript. Desta forma este projeto irá atender em maior parte os requisitos para serem usados por um banco digital, desenvolvido em uma arquitetura onde podemos ter uma fácil manutenibilidade, com novas tecnologias e evoluir facilmente de acordo com as necessidades que poderão surgir.

***Palavras-Chave**: Banco Digital, ReactJS, Javascript, autenticação.*

ABSTRACT

With the increase of the population with internet access, mainly using mobile devices, there is a need for banking applications that aim to facilitate banking processes in order to improve the user's life. In this way, digital banks emerge offering increasingly better products and services, reducing the monopoly of the large traditional banks and increasing competition. So this work is a software project that simulates a digital bank application, where the main focus is account control to perform traditional operations, such as transfers, payments, history, deposits, taking into account the entire security structure to provide a reliable service. This work presents the requirements, implementation and documentation of a digital bank application, for the implementation the ReactJs framework was used for the construction of interfaces, Javascript programming language. In this way, this project will mostly meet the requirements to be used by a digital bank, developed in an architecture where we can have easy maintainability, with new technologies and easily evolve according to the needs that may arise.

***Keywords**: Digital bank, ReactJS, Javascript, authentication.*

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE SIGLAS

API	*Application Programming Interface*
BACEN	Banco Central do Brasil
BCBS	Comitê de Supervisão Bancária da Basiléia
CPU	*Central Processing Unit*
CSU	Caso de Uso
DD	Dicionário de Dados
HTML	Linguagem de Marcação de Hipertexto
IDE	Ambiente de desenvolvimento integrado
MVC	Modelo, Visualização e Controle
PIX	Modelo de Transferência
PROER	Programa de Estímulo à Reestruturação e ao Fortalecimento do Sistema Financeiro Nacional
QR Code	*Quick Response Code*
RF	Requisitos Funcionais
RQ	Requisitos de Qualidade
RU	Requisitos de Usuários
SMS	Serviço de mensagem curta
TED	Transferência Bancária

SUMÁRIO

1 INTRODUÇÃO

Com o avanço da internet e acessos aos aplicativos *mobile,* verificou-se um crescente desenvolvimento tecnológico no setor financeiro, ampliando os acessos aos *smartphones* para fazer operações no setor bancário, facilitando o dia a dia do usuário através da desburocratização dos processos bancários como criar uma conta, alterar endereço, enviar dinheiro, extrato, entre outros.

De acordo com o Banco Central do Brasil (BACEN), o mercado bancário brasileiro vem crescendo desde 1980, as concorrências entre os bancos fazem o mercado bancário avançar em tecnologia otimizando os processos, segurança e relacionamento com os clientes. Cita ainda que este desenvolvimento teve impulso por motivos de desordem na economia, como por exemplo a alta inflação, desvalorização da moeda nacional frente ao dólar e descontrole fiscal (BACEN, 2019).

Na década de 90 houve a necessidade de mudanças vindo pelo BACEN. Para se firmar regras e limites para as instituições financeiras e bancárias, foram introduzidas normativas, mudanças legais e implantação de regras.

Essas mudanças foram necessárias em prol da modernização do sistema financeiro nacional por meio do Programa de Estímulo à Reestruturação e ao Fortalecimento do Sistema Financeiro Nacional (PROER).

Várias abordagens têm sido usadas pelas empresas para facilitar a vida dos clientes e promover uma visão sustentável das operações no meio ambiente, e levar a relação com o cliente para o mundo digital é uma dessas abordagens. Essa abordagem está intimamente ligada ao fato de que o envolvimento da população com tecnologia chegou a patamares sem precedentes na história. Segundo a pesquisa feito em 2017, 48% da população mundial possuem acesso à internet. Entre os jovens com idade entre 15 e 24 anos, esse número chega a 71% (ITU, 2017).

Os bancos digitais, cuja movimentação ocorre exclusivamente por meio eletrônico, surgem no contexto dos avanços tecnológicos dos últimos anos, que impactaram o setor bancário, tornando possíveis a abertura e o encerramento de contas, a realização de transferências, o acesso a linhas de crédito e a investimentos sem que o cliente precise em momento algum ir até uma agência bancária. Esse

modelo de operação se apresenta desde o início como um desafio para as instituições já estabelecidas, para as entrantes e para os órgãos reguladores.

Para os bancos, essas mudanças tecnológicas representaram uma redução dos custos de coleta, processamento e uso das informações, o que lhes permitiu otimizar inúmeros processos, incluindo o cálculo de riscos, de crédito e de custos, por exemplo.

Surgiram oportunidades de mercado para a criação de bancos digitais para trazer diversos benefícios ao usuário, descentralizando o monopólio dos grandes bancos no brasil oferecendo produtos e inovações de processos.

Os Bancos Digitais exploraram o nicho de mercado de clientes descontentes com o tempo perdido para ir até uma agência bancária e com baixa percepção de benefício em relação às tarifas e juros cobrados pelos bancos. Ao identificar essas lacunas do mercado, os Bancos Digitais causaram sérios questionamentos dos clientes às práticas até então constituídas (BORGES; FRANK, 2019).

Neste sentido, a disponibilização em canais digitais de produtos e serviços, que antes só poderiam ser acessados presencialmente, passou a ser um elemento estratégico essencial para os bancos tradicionais, buscando melhorar a percepção de atendimento dos clientes, desenvolvendo a sua lealdade e gerando maior rentabilidade. Esse desafio tem como premissa que manter um cliente leal é mais fácil e custa menos do que atrair novos consumidores (ENGEL; BLACKWELL; MINIARD,2000).

Com base na necessidade de atender essas novas empresas de banco digitais, este trabalho visa explorar as principais tarefas que são utilizadas em um aplicativo comum bancário utilizando ferramentas de desenvolvimento ágeis e moderna. Além de poder ser utilizado com a possibilidade de implementar outras soluções que possa vir aparecer, através de um sistema com fácil manutenibilidade.

Para atingir o objetivo, pretende-se desenvolver e documentar um aplicativo bancário utilizando ferramentas de desenvolvimento modernas, para fornecer algumas operações comuns de um aplicativo bancário tradicional. Utilizando tecnologias recentes e inovadoras fornecendo uma maior flexibilidade no software como manutenção, expansão, agilidade.

Para desenvolver esta aplicação é necessário elucidar os requisitos funcionais e de qualidade do sistema. Utilizaremos a linguagem de programação *Javascript* usando o *framework React Native*. A modelagem do software é de fundamental

importância então foram implementados os diagramas de caso de uso e a documentação do aplicativo.

Considerando todas as necessidades mostradas até aqui, este aplicativo se mostra importante com o objetivo de atender este novo mercado de bancos digitais, e mostrar o poder das novas linguagens de programação de ser capaz de produzir aplicativos de grande complexidade com equipe e tempo reduzido.

Linguagens nativas para mobile utilizam uma maior equipe de programadores e um alto nível de complexidade para implementar serviços, o que pode se tornar inviável pelo alto custo, e com a alta defasagem de profissionais programadores.

Com os recursos existente das novas linguagens de programação, este processo de implementar um aplicativo bancário se mostrou possível com apenas uma pessoa no front-end, recursos esses que visam facilitar e entregar agilidade para o programador.

A comunidade de programadores especificamente na linguagem *Javascript* fornece diariamente bibliotecas com novos recursos para otimizar os aplicativos.

Vale destacar a importância das disciplinas de Programação, Algoritmo e Paradigmas de Programação, que foram fundamentais para o entendimento da linguagem Javascript e a implementação no código, bem como sua estrutura. A manipulação das funções, variáveis, classe, atributos, entre outros conceitos que foram de suma importância para a construção deste trabalho.

Destacamos também a importante contribuição da disciplina de Engenharia de Software. Conhecimento que foi utilizado para fazer a documentação, foi utilizado caso de uso, levantamento de requisitos, diagrama de caso de uso.

Uma vez que este trabalho se mostra relevante, foi feito uma pesquisa aplicada, foi utilizado artigos atuais, leis bancárias e manuais das ferramentas de software que foram utilizadas para a execução do trabalho.

Este trabalho está dividido da seguinte forma: no capítulo 2 será mostrado o referencial teórico acerca de termos e ferramentas utilizadas, no capítulo 3 será mostrado a visão geral do desenvolvimento do aplicativo, sendo apresentado interfaces, restrições, limitações, especificações técnicas. Já o capítulo 4 apresenta os resultados e implementação do aplicativo e por fim, no capítulo 5 estão as considerações finais, dificuldades encontradas durante o desenvolvimento e sugestões para trabalhos futuros.

2 REFERENCIAL TEÓRICO

Neste capítulo são apresentadas as tecnologias utilizadas para o desenvolvimento do aplicativo, tal como a linguagem, o framework, o ambiente de programação e algumas funcionalidades como autenticação e reconhecimento de face.

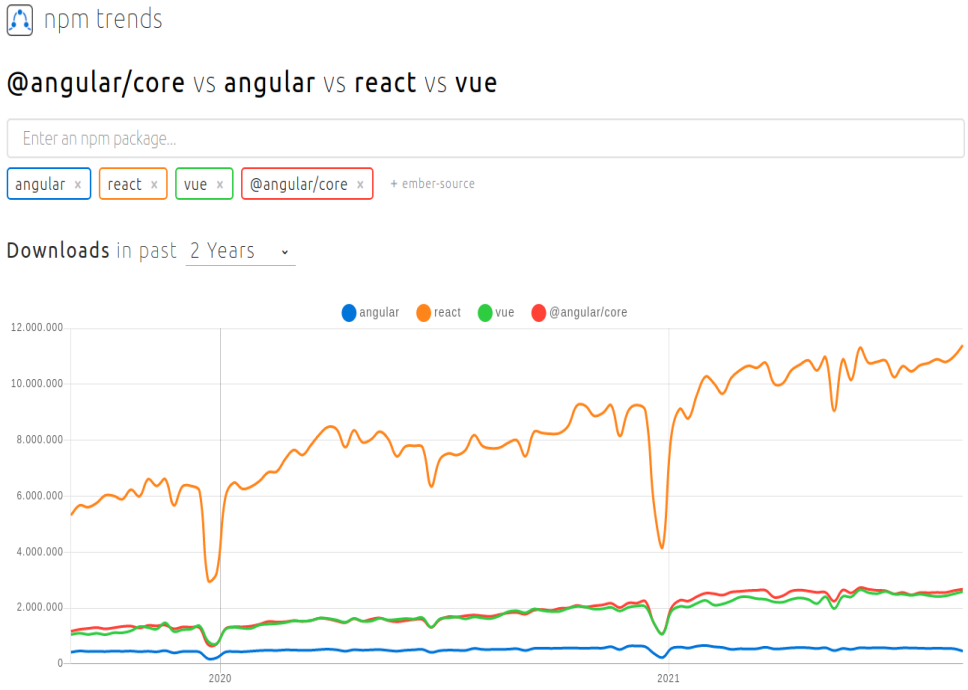
2.1 *Framework React*

React é uma biblioteca do *Javascript* que tem como objetivo construir interface de usuário (UI). Assim como existem outras bibliotecas, por exemplo, *Vue* e *Angular* também bibliotecas *Javascript*. É importante destacar a grande popularização do *React* desde o seu lançamento feito em 2013 pelo *Facebook*, quando disponibilizou o código aberto do *framework.* Desde então conforme a Figura 1 pode-se perceber a alta quantidade de *download* em relação às outras bibliotecas, nos anos de 2019 e 2020.

O *American Express*, *Netflix*, *Airbnb*, dentre outras grandes companhias utilizam o *React*, isso ajudou na popularização, com as frequentes atualizações do *Facebook*, seu criador, fez que resultasse no tamanho sucesso. Segundo o site do *React* sobre a definição, entende-se que o "*React* é uma biblioteca *Javascript* declarativa, eficiente e flexível para criar interfaces com o usuário. Ele permite compor interfaces de usuário complexas a partir de pequenos e isolados códigos chamados componentes" (TUTORIAL, 2019).

Mas qual a vantagem de ser uma linguagem declarativa? O paradigma de programação declarativa tem vantagens em relação a imperativa, onde precisa realmente escrever o código que define como um programa deve realizar tarefas. Por sua vez a linguagem declarativa utiliza menos código para realizar um objetivo por não ter que detalhar todas as ações da função de como ela irá se comportar, economizando tempo e o esforço, já que muitas funcionalidades fazem parte do sistema.

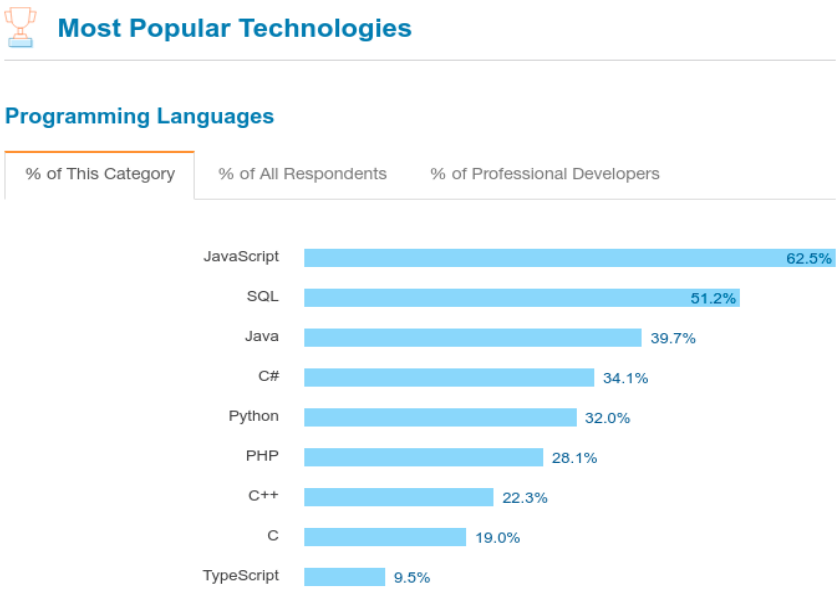
Figura 1 - Número de downloads Angular, *React*, Vue e Angular/Core



Fonte: npmtrends, 2021.

É importante ressaltar que a popularização do *React* também se justifica por utilizar a linguagem *Javascript* que segundo a *Developer Survey Results* 2021 (STACK OVERFLOW, 2021) é a linguagem mais utilizada pelos profissionais de desenvolvimento, conforme mostrado na Figura 2.

Figura 2 – Linguagens mais utilizadas por profissionais de desenvolvimento.



Fonte: Stack Overflow, 2021

2.2 Funcionamento do *React*

Conforme foi apresentado, sabe-se que o *React* é utilizado na parte visual da aplicação e tem como objetivo tornar a experiência de usuário mais eficiente, deste modo, é importante entender o que seria a parte visual da aplicação, explicando o padrão de projeto *Model-View-Controller*(MVC).

O padrão de arquitetura MVC tem como objetivo facilitar a troca de informações entre a interface do usuário e o banco de dados, tendo a camada de controle para intermediar essas ações de requisições. Essa arquitetura proporciona ao desenvolvedor uma manutenção mais fácil e reaproveitamento de código, produtividade, uniformidade na estrutura do software, melhora a documentação e reduz tempo de desenvolvimento de um projeto.

As camadas do MVC são responsabilidades e isolamento de regras de negócio da lógica da apresentação, isso possibilita a criação de diversas telas sem afetar a regra do negócio, proporcionando flexibilidade e reuso.

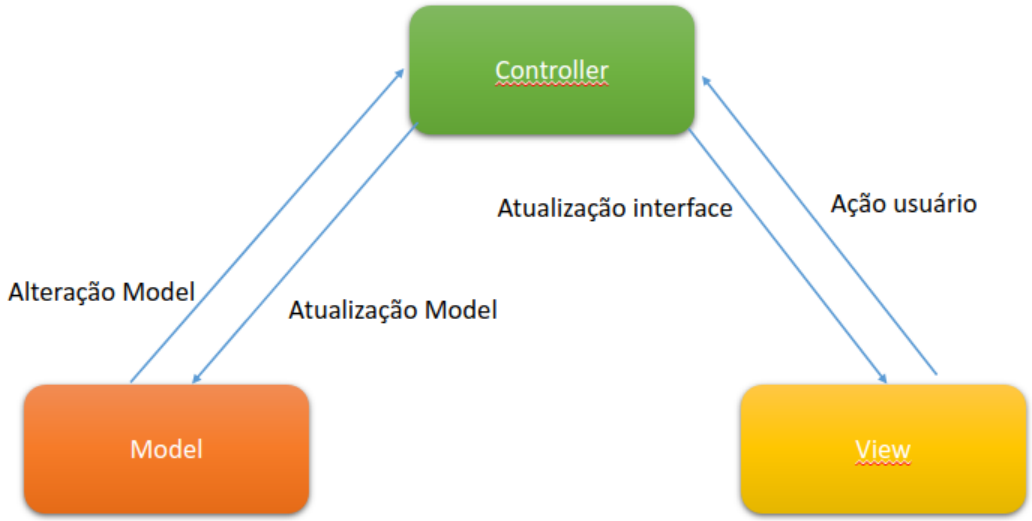
A camada de visualização (*View)* é responsável por apresentar as informações ao usuário, gerenciar as informações que são exibidas em tela. Interações feita pelo usuário e requisições são de responsabilidades da camada de visualização.

A camada de modelo (*Model)* é responsável por manipular os dados da aplicação, gerenciar e controlar a forma que os dados se comportam por meio das funções, lógicas e regras de negócio estabelecidas, recebe informações do *controller* e envia a resposta mais adequada.

A camada de controle (*Controller*) é responsável por interpretar as solicitações do usuário, que podem ser feitas por mouse, teclado e outros meios de interação. Após receber as interações ele verifica qual ação tomar, processa a informação e repassa para o *Model* ou para a *View*.

A Figura 3, demonstra tal arquitetura, que pode ser utilizada na programação web, mobile e desktop. Por conta dessas facilidades que o MVC oferece, passou a ser utilizado por diversos framework, tornando se o mais popular no desenvolvimento web.

Figura 3 - Estrutura do MVC.



Fonte: Elaborado pelo autor.

Entendendo esse conceito de MVC, podemos falar sobre a camada que utilizaremos no *React.* A camada *View* será a responsável por mostrar a interface para o usuário (UI). O *React* por ser uma biblioteca *front-end* utilizada para construir interface, fará requisições para *controllers* de *WebServices* recebendo e enviando dados via *Javascript Object Notation* (JSON).

JSON é um acrônimo para "*Javascript Object Notation*", é um formato de texto para a serialização de dados estruturados. JSON pode representar quatro tipos de dados primários: *strings*, números, booleanos e nulos e estruturados como objetos e vetores. Assim o JSON foi feito com o objetivo de ser simples, portável, textual.

Feito para fazer troca de dados, capaz de proporcionar um alto nível de interoperabilidade na troca de dados, isso é possível por meio da padronização de uma gramática simplificada que pode ser traduzida por qualquer linguagem.

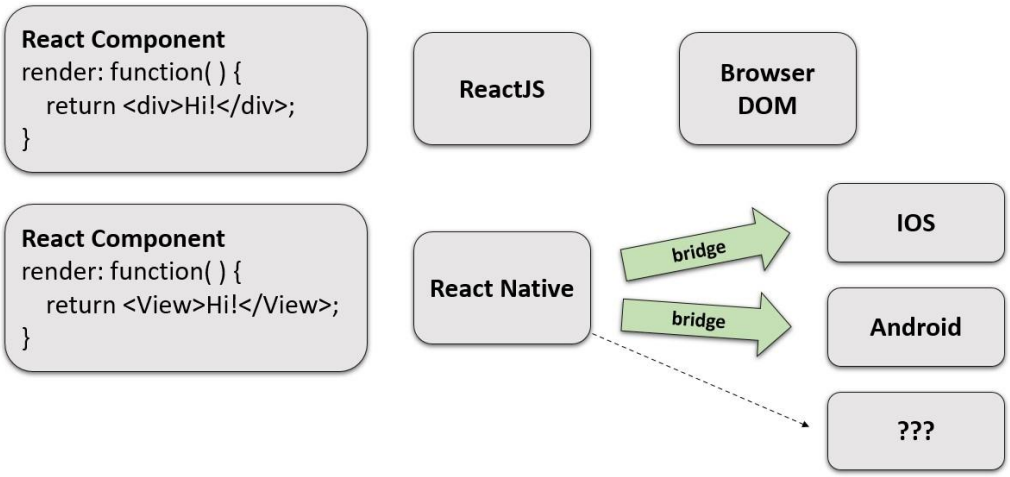
2.3 Framework React Native

Segundo Occhino (2015), engenheiro diretor do grupo *React* do Facebook©, o lema: “Aprenda uma vez, escreva em qualquer lugar”, significa que o desejo de fazer com que a experiência que os desenvolvedores têm para desenvolver para Web com o *ReactJS*, seja a mesma para desenvolver *mobile* (Figura 04).

Atualmente, há uma infinidade de opções para se desenvolver o mesmo projeto para diferentes plataformas, porém, cada aplicação tem suas peculiaridades e capacidades.

Seguindo o contexto acima, em 2015 o *Facebook*© lançou o *React Native*, *framework* para desenvolvimento mobile, inicialmente com suporte para iOS, e depois para Android. Destaca-se que basta desenvolver um app, que o mesmo poderá ser executado nas duas plataformas, Android ou iOS (Figura 04).

Figura 4 - Funcionamento do *react* em *Android* e iOS.



Fonte: Eisenman (2016).

O *React Native* é uma derivação do *React*, que por sua vez é uma biblioteca Javascript, feita para construir aplicativos móveis reais e nativos para IOS e Android. Sendo assim o desenvolvedor tem a possibilidade de usar a mesma biblioteca no desenvolvimento web e em plataformas móveis, e os aplicativos podem ser desenvolvidos de forma simultânea para as plataformas iOS e Android, o que facilita no tempo de desenvolvimento.

Com o *React Native*, a aplicação e toda lógica interna a ela, é escrita e executada em *Javascript*, o que possibilita ter um UI totalmente nativo. Os eventos são controlados: como o "estado" muda com o tempo e como os dados são preparados para exibição.

Assim como o ReactJS, o *React Native* é feito para trabalhar com componentes, assim permite o reuso do mesmo, pois podemos separar a tela de visualização em

pequenos componentes, cada qual com seu estilo e funcionalidade, sendo assim a manutenção posteriormente poderá ser feita apenas em uma parte e não influenciando em toda tela, garantido flexibilidade e segurança.

O *React Native* tem uma sintaxe semelhante à do *ReactJs*, pois também utiliza o JSX. JSX é uma extensão de sintaxe para o Javascript, permitindo que combine HTML com Javascript. Tal combinação simplifica toda a estrutura de codificação escrita de um site.

Apesar de usar a mesma sintaxe que o *ReactJs*, seus elementos não são renderizados da mesma forma. Como dito anteriormente, no *React Native* os elementos são renderizados de forma nativa, utilizando o Javascript Core, como uma ponte entre o JSX e as linguagens. Essa ponte abstrai uma camada de aplicação que possibilita executar API de renderização do Java e do Object-C (ALVES, 2019).

API provem do inglês *Application Programming Interface*, é um conjunto de rotinas e padrões estabelecidos por um software para a utilização das suas funcionalidades, por aplicações que não pretendem envolver nos detalhes da aplicação, apenas em usar o seu serviço.

React Native possui diversas funcionalidades que aumentam a praticidade e a produtividade. Uma delas que é importante destacar é o *Hot Reloading*, que faz com que o programa fique rodando em desenvolvimento, e a cada atualização no código uma versão nova é injetada na aplicação, levando menos de um segundo para atualizar. Vale ressaltar a possibilidade de depurar a aplicação pelo Google Chrome, como se fosse uma aplicação web, usando o recurso Dev *Tools*.

2.4 Javascript

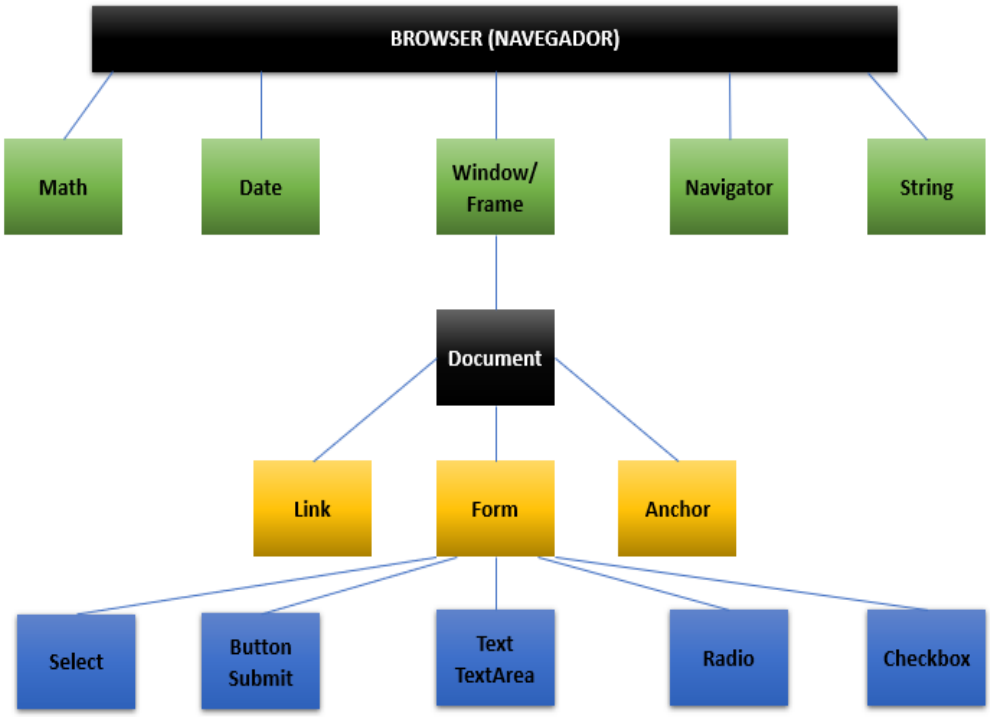
A linguagem *JavaScript* foi criada pela Netscape Communications Corporation4 e foi desenvolvida com o nome Mocha, depois passou a se chamar *LiveScript* e foi finalmente lançada como JavaScript em 1995 integrando a versão 2.0B3 do navegador Netscape. A linguagem foi criada para trabalhar com aplicações interativas nas páginas HTML (MOZILLA, 2021).

Os comandos *Javascript* são embutidos nas páginas e interpretados pelo navegador, ou seja, *Javascript* é uma linguagem interpretada. Foi submetida a uma norma internacional, que originou a especificação ECMA-262, que determina o padrão para a linguagem Javascript, que ficou conhecida como ECMAScript.

No *Javascript* todos os elementos na web são tratados como objetos. Estes objetos são agrupados de acordo com seu tipo e finalidade. A linguagem permite que os desenvolvedores possam criar objetos de acordo com a necessidade. Ao carregar uma página no navegador, objetos padrão são carregados automaticamente, sendo eles: *window, location, history, document*.

A linguagem manipula vários tipos de objetos através de suas propriedades e métodos. Esses objetos seguem uma hierarquia, fazendo que um objeto seja propriedade de outros, sendo chamados de propriedades, como podemos ver na Figura 5.

Figura 5 - Funcionamento do *Javascript* no navegador



Fonte: Elaborado pelo autor.

Javascript é uma linguagem completa e poderosa que possui muitas das qualidades de diversas outras linguagens, como: listas associativas, tipagem dinâmica e expressões regulares de Perl e a sintaxe similar a C/C++, linguagens de grande reconhecimento tanto no mundo acadêmico quanto comercialmente. Além disso, Javascript é multiparadigma e entre eles destacam-se a programação estrutural e orientada a objeto. (FLANAGAN, 2013).

2.5 Visual *Studio Community*

O *Visual Studio* foi criado pela Microsoft e pode ser definido como uma das melhores opções de IDE do mercado, especialmente para o desenvolvimento de aplicações baseadas na tecnologia .NET, onde oferece uma suíte de ferramentas integradas.

O Visual Studio é uma IDE, que no inglês significa *Integrated Development Environment* ou ambiente de desenvolvimento integrado, que é um programa de computador que reúne características e ferramentas de apoio ao desenvolvimento de software com o objetivo de agilizar este processo.

Por ser uma IDE gratuita, tem quase todos os recursos para proporcionar um melhor ambiente de desenvolvimento. Um ponto que vale ressaltar é por ele ser multiplataforma, podendo ser utilizado no Mac, Linux e *Windows*, também pode ser utilizado com diversos idiomas, personalizável e com diversos plugins feitos pela própria comunidade.

2.6 Autenticação

Autenticar é estabelecer ou confirmar algo como autêntico, uma confirmação que reivindica autoria de algo, geralmente relacionada a identidade. Existem várias formas de se fazer uma autenticação. As mais conhecidas são através de usuário e senhas, biometria, carteira de identidade facial, certificado digital, autenticação por sessão, autenticação por token, entre outras.

No contexto deste aplicativo, o fator principal é a segurança, a autenticação do usuário para acessar o aplicativo e verificar se é o próprio usuário que está usando as funcionalidades do aplicativo é uma preocupação clara, onde é preciso minimizar as falhas possíveis que possa ocorrer, então utilizamos os principais recursos disponíveis hoje para autenticação, utilizando a biometria: reconhecimento fácil e impressão digital para liberação das funcionalidades do aplicativo.

Biometria é utilizada para reconhecimento, como cada pessoa é única, então possuímos características únicas, que podem ser utilizadas como recurso de segurança. Essas características podem ser por voz, retina, íris dos olhos e palma da mão.

Os principais modelos de biometria utilizados para recurso de segurança, são elas: impressão digital; Geometria da mão; Reconhecimento de assinatura; Reconhecimento de íris; reconhecimento de retina; reconhecimento de voz. Em nossa aplicação utilizaremos a biometria por impressão digital e reconhecimento de íris.

2.6.1 Impressão digital

O recurso de biometria para permissão no aplicativo em conjunto, consegue nos proporcionar muita segurança. Assumindo que o acesso ao *smarthphone* é seguro o suficiente, o uso da biometria nos traz seguranças mesmo em casos de roubo e perda do dispositivo.

Ao cadastrar a digital no *smarthphone*, a imagem é encriptada e transformada em código binário, que age como uma camada de segurança, para as autenticações posteriores, os algoritmos de detecção de minúcias leva a fazer essa análise que serão comparadas com digital gravada anteriormente.

As minúcias ou pontos característicos são acidentes que se encontram nas cristas papilares como, por exemplo, linhas de determinam abruptamente ou se bifurcam, e tem por finalidade estabelecer a unicidade das impressões digitais. O *American National Standards Institute* (ANSI) propôs quatro maiores grupos de minúcias: cristas finais, bifurcações, cruzamentos e pontos indeterminados. Porém, as minúcias consideradas mais importantes são as cristas finais e bifurcações, pois ocorrem frequentemente nas imagens de impressões digitais (SANTOS, 2016).

Figura 6 - Impressão digital do polegar

Fonte: Santos (2016).

A verificação da impressão digital tende a ser seguro desde que haja coincidência em no mínimo doze pontos característicos e que não haja nenhuma discordância entre estes pontos. Precisam ser idênticos e ter a mesma localização.

À vista disso este recurso é fundamental para segurança do aplicativo, e fornecer segurança aos usuários.

2.6.2 Reconhecimento facial - *Facematch*

A biometria por reconhecimento facial que é um dos processos mais utilizados quando se trata de autenticação. Consegue identificar rapidamente qualquer pessoa afim de oferecer segurança para determinados acessos, este recurso tem sido implementado por diversos sistemas por se mostrar confiável e eficiente.

Por se tratar de um aplicativo que lida com dados sensíveis, é primordial atender todos os pontos de segurança, o *Facematch* por sua vez contribui com este pilar de segurança. Esta ferramenta consiste em fazer uma comparação de característica e elementos visuais de uma imagem de rosto que foi feito a captura com uma imagem arquivada no banco de dados.

A comparação não é feita em tempo real. A imagem de captura é enviada para a ferramenta que trará um retorno verdadeiro ou falso de acordo com a comparação feita. Então para usar esta ferramenta é necessário o cadastro prévio do usuário no banco de dados para fazer a comparação.

Este recurso consiste em processar a imagem que é retirada por meio da câmera do dispositivo. Essa imagem é extraída a fim de fornecer informações visuais, como feições estruturais. O pré-processamento envolve técnicas de realce de contrastes e remoção de ruídos.

Para fazer o reconhecimento após processar a imagem, são utilizadas duas técnicas para melhorar a qualidade de entrada: filtragem de ruido e normalização da iluminação. Com a filtragem do ruido, consegue-se eliminar variações na imagem que é feita no momento da aquisição, frequentemente causada pelo sensor de captura. A normalização da iluminação é responsável por amenizar as variações na imagem causada por diferentes fontes de iluminação no momento da captura.

Sabendo da importância da ferramenta, foi implementado em nosso aplicativo para autenticação do usuário, como um dos recursos de segurança.

Na data atual deste trabalho, temos várias empresas que ajudam este processo de autenticação, fornecem APIs que fazem este serviço de reconhecimento e nos devolve uma resposta, desta maneira conseguimos facilitar o processo de desenvolvimento e ter uma entrega mais rápida sem prejudicar a segurança.

Em nossa aplicação foi utilizado uma dessas empresas que armazenam a foto da biometria digital e de reconhecimento facial, assim que é feito o cadastro de um novo usuário, essas imagens são utilizadas posteriormente para validação de acesso e permissões do aplicativo. Para o processo de reconhecimento seja por facial ou por impressão digital, após a foto é enviado o binário para a API da empresa terceirizada que irá fazer a validação entre a foto de cadastro com a nova foto enviada e nos devolve um *true* ou *false*. Com essa resposta podemos prosseguir o acesso ou barrar informando uma mensagem de erro.

3 DESENVOLVIMENTO

Para aplicar o conceito do *framework React native* foi feito a análise e documentação do aplicativo, para criar um modelo de aplicativo para um banco digital.

3.1 Aspecto geral do aplicativo

O aplicativo foi construído com a linguagem *Javascript* utilizando o *framework React Native* com o objetivo de fornecer um produto completo para atender um novo banco digital.

3.2 Interfaces do aplicativo

O aplicativo é composto das seguintes interfaces gráficas:

- Tela de login: Nesta tela contém os campos de CPF e senha para acesso a conta;
- Tela de cadastro: Nesta tela contém CPF, nome completo e número de celular para ser inseridos;
- Tela de recuperação de senha: Contém os campos para preenchimento das informações necessárias para a recuperação da senha do usuário;
- Tela inicial: Contém informações da conta como:
    - Nome do usuário autenticado;
    - Dados da conta do usuário;
    - Saldo disponível na conta;
    - Ícone para fazer logoff
    - Botões de transferência, extrato, cobrança e pagamentos.
- Tela de transferência: Cadastrar uma nova conta para eventualidade de transferência e pesquisar as contas cadastradas.
- Tela de cobrança: digitar valor da cobrança e definir o tipo da cobrança como:
    - Cobrança por PIX QR Code;
    - Cobrança por boleto.
- Tela de extrato: contém dois intervalos de datas para definir o período do extrato a ser mostrado.

- Tela de pagamentos: O pagamento de um boleto é por via código de barras. Ou digitando o código, ou captura usando a câmera do celular para identificar o código de barras.

3.3 Interface do usuário

O sistema foi desenvolvido para se adaptar a diferentes resoluções e tamanhos de tela, podendo ser utilizado com internet, com sistema operacional Android ou iOS.

3.4 Requisito de hardware

O software foi desenvolvido para os requisitos mínimos:

- Processador: QuadCore (1.8GHz);
- Memória RAM: 4 GigaBytes;
- Armazenamento livre: 100 Megabytes de armazenamento livre.

3.5 Sistemas operacionais compatíveis

O aplicativo pode ser instalado em dispositivos iOS e Android, tendo como requisitos mínimos as versões:

- Android Jelly Bean, v16, 4.1.x ou mais recente;
- iOS versão 8.0 ou superior.

3.6 Características dos usuários

Usuários que irão se cadastrar no banco buscando praticidade para fazer movimentações financeiras, assim como outras operações e benefícios que um banco digital pode oferecer.

3.7 Funções do sistema

Esta seção apresenta as principais funções que o sistema realizará:

- Realizar cadastro no aplicativo;

- Realizar login com duas opções de validações; biometria e *facematch*;
- Permitir recuperação de senha via e-mail;
- Visualizar a tela principal do usuário como saldo e número da conta;
- Visualizar o botão de transferir dinheiro, cobrar e efetuar pagamentos;
- Visualizar o botão de mostrar extrato.

3.8 Restrições/Limitações

O aplicativo possui as seguintes restrições:

- O sistema só pode ser acessado quando o usuário tiver uma conexão com a internet;
- O usuário só poderá fazer movimentações como transferência de acordo com limites e horários pré-estabelecidos pelo administrador.

4 DOCUMENTAÇÃO DO APLICATIVO

Este tópico apresenta a documentação utilizada para o desenvolvimento do aplicativo de um banco digital. Esta documentação inclui: Requisitos de usuários - Necessidades (RU), Requisitos Funcionais (RF), Requisitos de qualidade (RQ) e Casos de Uso descritivos (CSU).

4.1 Requisitos de Usuários - Necessidades

Esta seção apresenta os requisitos de usuário – necessidades definidas no processo de levantamento de requisitos.

Quadro 1 - Requisitos de Usuários - Necessidades

ID	Descrição
RU 001	Usuário faz cadastro no sistema.
RU 002	Usuário faz login no sistema.
RU 003	Usuário recupera dados de login.
RU 004	Usuário visualiza a tela principal.
RU 005	Usuário poderá realizar transferência.
RU 006	Usuário poderá efetuar pagamentos.
RU 007	Usuário poderá retirar extrato.
RU 008	Usuário poderá fazer boleto de cobrança.

Fonte: Elaborado pelo autor.

4.2 Requisitos funcionais

Esta seção apresenta os requisitos funcionais definidos no processo de levantamento de requisitos.

Quadro 2 - RF 001: Cadastrar Usuário

Identificador RF 001	**Nome** Cadastrar Usuário
Caso de uso	**Autor**

CSU 001	Reverson Thaiam
Descrição O sistema deve permitir o cadastro do usuário que deverá informar os dados pessoais.	
Critério de verificação • Verificar se o e-mail é valido; • Verificar se o CPF ou CNPJ é valido; • Verificar se o número de telefone é valido; • Verificar se o número do CEP é valido;	
Dependência RU 001, RF 001	**Prioridade** Essencial

Fonte: Elaborado pelo autor.

Quadro 3 - RF 002: Realizar login

Identificador RF 002	**Nome** Realizar login
Caso de uso CSU 002	**Autor** Reverson Thaiam
Descrição O sistema deve permitir que o usuário informa o CPF ou CNPJ e senha.	
Critério de verificação • Verificar se o CPF ou CNPJ é valido. • Validar senha ou biometria digital. • Validar o usuário por reconhecimento facial.	
Dependência RU 002, RF 001	**Prioridade** Essencial

Fonte: Elaborado pelo autor.

Quadro 4 - RF 003: Recuperar senha.

Identificador RF 003	**Nome** Recuperar senha.
Caso de uso CSU 003	**Autor** Reverson Thaiam

Descrição	
O sistema deve permitir que o usuário recupere sua senha. Para isso deve informar o CPF ou CNPJ cadastrado e será enviado uma senha temporária para o celular cadastrado.	
Critério de verificação	
Dependência	**Prioridade**
RU 003, RF 001	Essencial

Fonte: Elaborado pelo autor.

Quadro 5 - RF 004: Visualizar tela principal.

Identificador	**Nome**
RF 004	Visualizar tela principal.
Caso de uso	**Autor**
CSU 004	Reverson Thaiam
Descrição	
O sistema deve mostrar as informações como: • Nome da conta; • Número da conta; • Saldo da conta; • Botão de transferência; • Botão de extrato; • Botão de pagamentos; • Botão de boletos.	
Critério de verificação	
• Verificar se o usuário foi autenticado anteriormente.	
Dependência	**Prioridade**
RU 004, RF 002	Essencial

Fonte: Elaborado pelo autor.

Quadro 6 - RF 005: Visualizar tela de transferência.

Identificador	**Nome**
RF 005	Recuperar senha.
Caso de uso	**Autor**
CSU 005	Reverson Thaiam

Descrição	
O sistema deve mostrar em forma de lista as contas de usuário favoritos.	
Critério de verificação	
• Verificar se o usuário foi autenticado anteriormente.	
Dependência	**Prioridade**
RU 005, RF 002	Essencial

Fonte: Elaborado pelo autor.

Quadro 7 - RF 006: Fazer pagamentos

Identificador	**Nome**
RF 006	Fazer pagamentos de boleto.
Caso de uso	**Autor**
CSU 006	Reverson Thaiam
Descrição	
O sistema deve permitir efetuar pagamentos de boletos através do código de barras que poderá ser digitado por número ou pela câmera do dispositivo.	
Critério de verificação	
• Verificar se o usuário foi autenticado anteriormente.	
Dependência	**Prioridade**
RU 006, RF 002	Essencial

Fonte: Elaborado pelo autor.

Quadro 8 - RF 007: Mostrar extrato

Identificador	**Nome**
RF 007	Mostrar extrato.
Caso de uso	**Autor**
CSU 007	Reverson Thaiam
Descrição	
O sistema deve permitir digitar o intervalo de data que deseja retirar o extrato.	
Critério de verificação	
• Verificar se o usuário foi autenticado anteriormente.	
Dependência	**Prioridade**
RU 007, RF 002	Essencial

Fonte: Elaborado pelo autor.

Quadro 9 - RF 008: Cobrar valor

Identificador RF 008	**Nome** Cobrar valor.
Caso de uso CSU 008	**Autor** Reverson Thaiam
Descrição O sistema deve permitir gerar boletos para receber pagamentos, os boletos poderão ser gerados por código de barras ou QR Code PIX.	
Critério de verificação • Verificar se o usuário foi autenticado anteriormente.	
Dependência RU 008, RF 002	**Prioridade** Essencial

Fonte: Elaborado pelo autor.

4.3 Requisitos de qualidade

Esta seção apresenta os requisitos de qualidade definidos no processo de levantamento de requisitos.

Quadro 10 - RQ 001 Confidencialidade.

Identificador	RQ 001
Nome	Confidencialidade
Caso de Uso	
Descrição	As informações dos usuários são restritas e só podem ser acessadas e visualizadas pelo próprio.
Autor	Reverson Thaiam Batista dos Santos
Critério de verificação	
Dependência	
Prioridade	Essencial

Fonte: Elaborado pelo autor.

Quadro 11 - RQ 002 Usabilidade.

Identificador	RQ 002

Nome	Usabilidade
Caso de Uso	
Descrição	Os clientes do aplicativo devem conseguir usar o aplicativo sem precisar de ajuda.
Autor	Reverson Thaiam Batista dos Santos
Critério de verificação	
Dependência	
Prioridade	Essencial

Fonte: Elaborado pelo autor.

Quadro 12 - RQ 003 Confiabilidade.

Identificador	RQ 003
Nome	Confiabilidade.
Caso de Uso	
Descrição	O servidor deve ser confiável podendo suportar vários usuários simultâneos.
Autor	Reverson Thaiam Batista dos Santos
Critério de verificação	
Dependência	
Prioridade	Essencial

Fonte: Elaborado pelo autor.

Quadro 13 - RQ 004 Desempenho.

Identificador	RQ 004
Nome	Desempenho.
Caso de Uso	
Descrição	O aplicativo deve ter um bom desempenho, com os botões espaçados, com poucos carregamentos e ser responsivo.
Autor	Reverson Thaiam Batista dos Santos
Critério de verificação	
Dependência	
Prioridade	Essencial

Fonte: Elaborado pelo autor.

Quadro 14 - RQ 005 Manutenibilidade.

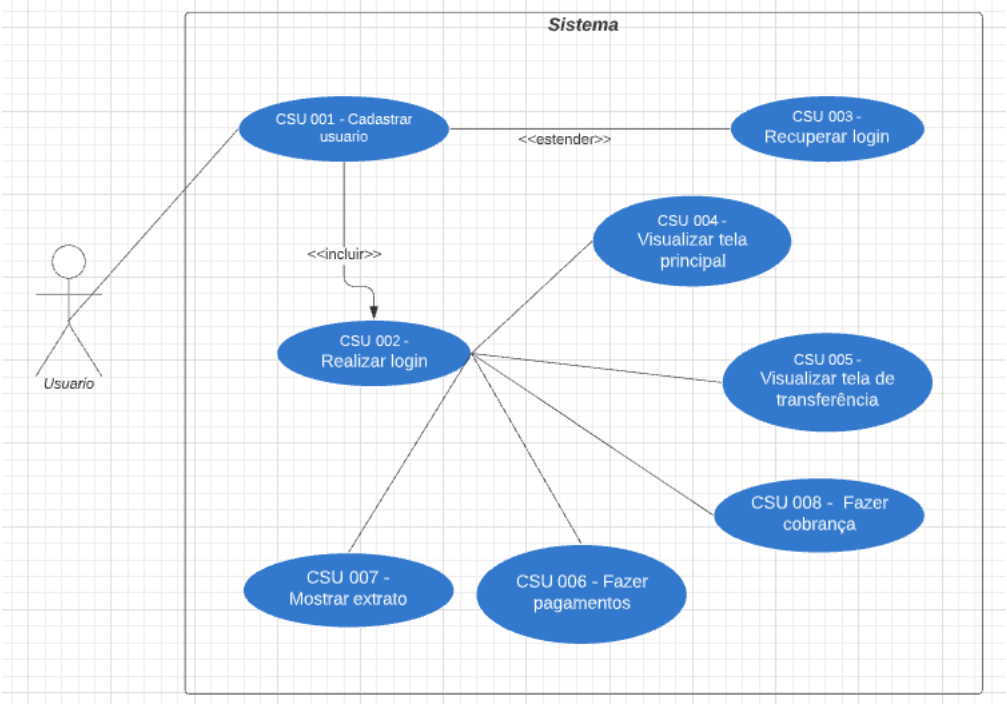
Identificador	RQ 005
Nome	Manutenibilidade
Caso de Uso	
Descrição	O aplicativo deve ser de fácil manutenção.
Autor	Reverson Thaiam Batista dos Santos
Critério de verificação	
Dependência	
Prioridade	Essencial

Fonte: Elaborado pelo autor.

4.4 Diagrama de casos de uso

Foi desenvolvido um diagrama de casos de uso para apresentar as funcionalidades do aplicativo, conforme a figura 7.

Figura 7 - Diagrama de casos de uso.



Fonte: Elaborado pelo autor.

4.5 Casos de uso descritivos

A seguir são descritos os casos de uso apresentados na Figura 1.

CSU 001: Cadastrar Usuário.

Identificador	CSU 001
Nome	Cadastrar Usuário
Atores	Usuário
Requisitos	RF 001
Responsável	Reverson Thaiam
Descrição/Resumo	Este caso de uso descreve o processo para o cadastro de um usuário.
Pré-condições	

Pós-Condições	
Cenário Principal	1. O usuário abre o aplicativo; 2. O sistema exibe a tela de login; 3. O usuário seleciona a opção “Cadastrar”; 4. O sistema exibe a tela de cadastro; 5. O usuário preenche os campos; 6. O usuário seleciona a opção “Próximo”; 7. O sistema exibe a tela registro de foto; 8. O usuário realiza uma foto do seu rosto; 9. O sistema exibe a tela solicitando a foto dos documentos: frente e verso; 10. O usuário faz uma foto frente e verso do documento; 11. O sistema exibe a tela de complemento do cadastro, solicitando informações pessoais; 12. O usuário preenche os campos 13. O sistema exibe a tela de complemento do cadastro, solicitando o endereço; 14. O usuário preenche os campos; 15. O usuário seleciona a opção "Concluir" 16. O sistema valida os dados informados; 17. O sistema envia os dados para o banco de dados para armazenamento e validação; 18. O sistema exibe uma tela informando "Análise pendente" 19. O usuário é redirecionado 20. Fim do caso de uso.
Cenários alternativos	**(A1) Passos 4 – O usuário cancela o cadastro** 1. A - O usuário seleciona a opção “Cancelar” ou fecha o aplicativo; 2. A - O sistema retorna ao passo 2 do cenário principal. **(A2) Passos 4 até o 15 – O usuário cancela o cadastro**

	1. B - O sistema cancela o cadastro sem concluir o cadastro; 2. B - O sistema retorna ao passo 2 do cenário principal.
Cenário de exceção	**(E1) Passo 4 – O usuário informa um e-mail já cadastrado no sistema** 1. A - O sistema apresenta a mensagem “E-mail já cadastrado”; 2. A - O sistema retorna ao passo 4 do cenário principal. **(E2) Passo 4 – O usuário informa CPF ou CNPJ já cadastrado no sistema** 1. B - O sistema apresenta a mensagem “CPF ou CNPJ já cadastrados”; 2. B - O sistema retorna ao passo 4 do cenário principal. **(E3) Passo 11 – O usuário informa dados pessoais errados ou não preenche todos os campos** 1. C - O sistema apresenta a mensagem “Preencha os dados e tente novamente”; 2. C - O sistema retorna ao passo 13 do cenário principal. **(E4) Passo 13 – O usuário informa dados de endereço errados ou não preenche todos os campos** 3. D - O sistema apresenta a mensagem “Preencha os dados e tente novamente”; 4. D - O sistema retorna ao passo 13 do cenário principal.
Qualidades	

Fonte: Elaborado pelo autor.

CSU 002: Fazer login.

Identificador	CSU 002
Nome	Fazer Login
Atores	Usuário
Requisitos	RF 002
Responsável	Reverson Thaiam
Descrição/Resumo	Este caso de uso descreve o processo para o login do usuário.
Pré-condições	
Pós-Condições	
Cenário Principal	1. O usuário abre o aplicativo; 2. O sistema exibe a tela de login; 3. O usuário preenche o CPF ou CNPJ e a senha; 4. O usuário faz o reconhecimento facial; 5. O sistema valida os dados informados 6. O usuário é redirecionado para a tela principal; 7. Fim do caso de uso.
Cenários alternativos	**(A1) Passos 5 e 6 – O usuário cancela o login** • O usuário fecha o aplicativo; • O sistema cancela o login; • O sistema retorna ao passo 7 do cenário principal. **(A2) Passos 3 e 4 – O usuário seleciona a opção de cadastro** • O usuário seleciona a opção “cadastrar”; • O sistema retorna ao passo 7 do cenário principal.
Cenário de exceção	**(E1) Passo 5 – O usuário informa um CPF não cadastrado no sistema ou uma senha diferente** • O sistema apresenta a mensagem “CPF ou senha inválidos”; • O sistema retorna ao passo 3 do cenário principal.
Qualidades	

Fonte: Elaborado pelo autor.

CSU 003: Recuperar Login.

Identificador	CSU 003
Nome	Recuperar Login
Atores	Usuário
Requisitos	RF 003
Responsável	Reverson Thaiam
Descrição/Resumo	Este caso de uso descreve o processo para recuperar dados de login do usuário.
Pré-condições	
Pós-Condições	
Cenário Principal	1. O usuário abre o aplicativo; 2. O sistema exibe a tela de login; 3. O usuário seleciona a opção “Esqueci minha senha”; 4. O sistema exibe a tela de recuperação de senha; 5. O usuário preenche os dados; 6. O usuário seleciona a opção “Confirmar”; 7. O sistema valida os dados informados; 8. O sistema envia um SMS com a nova senha para o telefone cadastrado; 9. O usuário acessa a caixa de SMS para obter a nova senha; 10. O sistema exibe a tela de senha que foi enviado no e-mail; 11. O usuário informa a senha do SMS e altera para uma nova senha; 12. O sistema exibe a mensagem “Senha alterada com sucesso”. 13. Fim do caso de uso.
Cenários alternativos	**(A1) Passo 4 – O usuário cancela a recuperação de senha** • O usuário seleciona a opção “Cancelar” ou fecha o aplicativo; • O sistema retorna ao passo 2 do cenário principal.

Cenário de exceção	**(E1) Passo 5 – O usuário informa CPF ou CNPJ inválido** • O sistema exibe a mensagem "CPF ou CNPJ não encontrado". • O sistema retorna ao passo 5 do cenário principal.
Qualidades	

Fonte: Elaborado pelo autor.

CSU 004: Visualizar tela principal.

Identificador	CSU 004
Nome	Visualizar tela principal
Atores	Usuário
Requisitos	RF 001
Responsável	Reverson Thaiam
Descrição/Resumo	Este caso de uso apresentará a tela principal
Pré-condições	
Pós-Condições	
Cenário Principal	1. O usuário abre o aplicativo; 2. O sistema exibe a tela de login; 3. O usuário faz login 4. O sistema exibe a tela com botão de pagamentos, depósitos, transferência e extrato.
Cenários alternativos	**(A1) Passo 4 – O usuário cancela o acesso** • O usuário fecha a tela do aplicativo.
Cenário de exceção	
Qualidades	

Fonte: Elaborado pelo autor.

CSU 005: Tela de transferência.

Identificador	CSU 005
Nome	Fazer transferência
Atores	Usuário
Requisitos	RF 005

Responsável	Reverson Thaiam
Descrição/Resumo	Este caso de uso descreve o processo para fazer transferência no sistema.
Pré-condições	
Pós-Condições	
Cenário Principal	1. O sistema mostra a tela principal com todas as funcionalidades, como: a. Transferir; b. Cobrar; c. Extrato; d. Pagamentos; e. *Logoff*. 2. O usuário seleciona a opção "Transferir"; 3. O sistema exibe a tela de transferências; 4. O sistema exibe a lista de contatos que foram cadastrados anteriormente; 5. O usuário deverá cadastrar uma conta ou selecionar uma conta já cadastrada; 6. Para cadastrar o usuário deverá selecionar se a conta é PIX ou TED no modelo de transferência; 7. O usuário deverá preencher os dados da conta informados pelo sistema; 8. O sistema irá salvar os dados preenchidos no banco de dados; 9. O sistema irá redirecionar o usuário para a tela de transferência solicitando o valor; 10. O usuário digita o valor e seleciona a opção "Próximo"; 11. O sistema exibe a tela para validação dos dados, como dados pessoais e informação da conta; 12. O usuário valida os dados e seleciona a opção "Próximo";

	13. O sistema exibe a tela solicitando biometria e senha da conta; 14. O usuário preenche os dados solicitados pelo sistema e seleciona a opção "Confirmar"; 15. O sistema faz a transferência e guarda as informações; 16. O sistema exibe a tela do comprovante com as informações da transferência; 17. O sistema exibe um botão para compartilhar o comprovante.
Cenários alternativos	**(A1) Passo 3 – O usuário cancela a transferência** • O usuário seleciona a opção “Cancelar” ou fecha o aplicativo; **(A2) Passo 7 – O usuário digita uma conta inválida** • O sistema não reconhece a conta; • O sistema solicita para digitar novamente. **(A2) Passo 10 – O usuário deixa o valor vazio** • O sistema exibe uma mensagem que o valor deve ser preenchido; • O sistema solicita para o usuário digitar novamente. **(A3) Passo 10 – O usuário não tem saldo suficiente** • O sistema exibe uma mensagem "Saldo insuficiente"; • O sistema solicita para o usuário digitar novamente. **(A4) Passo 12 – O usuário não aprova as informações** • O usuário volta cancela a operação; • O sistema retorna ao passo 1.

Cenário de exceção	**(A4) Passo 14 – O usuário informa dados inválidos** • O sistema mostra a mensagem "Senha invalida"; • O usuário preenche os dados novamente.
Qualidades	

Fonte: Elaborado pelo autor.

CSU 006: Fazer pagamentos.

Identificador	CSU 006
Nome	Fazer pagamentos
Atores	Usuário
Requisitos	RF 006
Responsável	Reverson Thaiam
Descrição/Resumo	Este caso de uso descreve o processo para fazer pagamentos no sistema.
Pré-condições	
Pós-Condições	
Cenário Principal	1) O sistema mostra a tela principal com todas as funcionalidades, como: a) Transferir; b) Cobrar; c) Extrato; d) Pagamentos; e) *Logoff*. 2) O usuário abre o botão de pagamentos; 3) O sistema abre a câmera do celular para fazer leitura do código de barras e oferece a opção para digitar manualmente o número; 4) O usuário faz a leitura do código de barras usando a câmera ou digitar manualmente o número do código de barras; 5) O usuário é encaminhado para digitar a senha e validar as informações do pagamento.

Cenários alternativos	**(A1) Passo 3 – O usuário cancela o pagamento** • O usuário seleciona a opção "<" para voltar a tela principal. **(A2) Passo 4 – O sistema não reconhece o código** • O sistema não reconhece o código do boleto na leitura; • O sistema continua na tela de leitura com a câmera ligada. **(A3) Passo 4 – O sistema não reconhece o número do código** • O sistema não reconhece o código do boleto digitado manualmente; • O sistema exibe uma mensagem "Boleto inválido"; • O sistema continua na tela 4.
Cenário de exceção	
Qualidades	

Fonte: Elaborado pelo autor.

CSU 007: Mostrar extrato.

Identificador	CSU 007
Nome	Mostrar extrato
Atores	Usuário
Requisitos	RF 001
Responsável	Reverson Thaiam
Descrição/Resumo	Este caso de uso descreve o processo para retirar um extrato no sistema.
Pré-condições	
Pós-Condições	
Cenário Principal	1. O sistema mostra a tela principal com todas as funcionalidades, como: a. Transferir;

	b. Cobrar; c. Extrato; d. Pagamentos; e. *Logoff*. 2. O usuário seleciona a opção "extrato"; 3. O sistema exibe a tela principal do extrato; 4. O sistema exibe dois botões para selecionar o intervalo das datas; 5. O usuário informa a data de início e data final; 6. O usuário seleciona o botão “mostrar extrato”; 7. O sistema exibe todas as operações dentro dessa faixa de data.
Cenários alternativos	**(A1) Passo 3 – O usuário cancela o extrato** • O usuário seleciona a opção “Cancelar” ou fecha o aplicativo; **(A2) Passo 7 – O usuário não tem extrato** • O sistema informa que o usuário não tem extrato na faixa de data.
Cenário de exceção	**(E1) Passo 5 – O usuário digita uma data inválida** • O sistema mostra uma mensagem para inserir a data novamente.
Qualidades	

Fonte: Elaborado pelo autor.

CSU 008: Fazer cobrança.

Identificador	CSU 008
Nome	Fazer cobrança
Atores	Usuário
Requisitos	RF 001
Responsável	Reverson Thaiam
Descrição/Resumo	Este caso de uso descreve o processo para retirar um boleto de cobrança no sistema
Pré-condições	

Pós-Condições	
Cenário Principal	1. O sistema mostra a tela principal com todas as funcionalidades, como: a. Transferir; b. Cobrar; c. Extrato; d. Pagamentos; e. *Logoff*. 2. O usuário seleciona o botão “cobrar”; 3. O sistema exibe a tela principal de cobrança; 4. O sistema solicita o valor a ser cobrado; 5. O usuário deverá selecionar o valor de cobrança; 6. O sistema pergunta se a cobrança irá ser feita por boleto ou por PIX QR Code; 7. O usuário seleciona o tipo da cobrança; 8. O sistema exibe um botão para compartilhar o código do boleto ou a imagem do QR Code.
Cenários alternativos	**(A1) Passo 3 – O usuário cancela a cobrança** • O usuário seleciona a opção “Cancelar” ou fecha o aplicativo; **(A2) Passo 4 – O usuário deixa o valor vazio** • O sistema exibe uma mensagem "Preencha o valor do boleto"; • O sistema continua no passo 4.
Cenário de exceção	
Qualidades	

Fonte: Elaborado pelo autor.

5 IMPLEMENTAÇÕES E RESULTADOS

Neste capítulo são apresentadas as telas do aplicativo do banco digital, descrevendo os seus componentes e detalhando as ações que podem ser realizadas pelos usuários, além de apresentar uma comparação entre a visualização dos dados nativa do *Oracle* e a visualização que o aplicativo oferece.

5.1 Tela inicial

Esta é a tela de apresentação do aplicativo ao usuário, Figura 8, na qual dá boas-vindas e pergunta se o usuário tem login.

Figura 8 – Tela inicial do aplicativo.

Fonte: Elaborado pelo autor.

5.2 Tela de Login

A figura 9 apresenta a tela inicial do aplicativo, onde o usuário tem a opção de realizar o login, recuperar a senha ou cadastrar-se. Essa tela é composta pelos componentes:

- CPF: Campo onde o usuário informa um CPF previamente cadastrado;
- Senha: Campo onde o usuário informa a senha correspondente ao CPF cadastrado;
- Entrar: Botão que ao ser pressionado irá realizar a validação dos campos de CPF e senha, verificando se estes campos correspondem aos dados cadastrados. Caso os dados sejam validados com sucesso, o usuário será redirecionado para a Tela Principal e, caso contrário, será exibida uma mensagem informando o erro;
- Esqueci minha senha: Botão que irá redirecionar o usuário para a Tela de Recuperação de Senha;

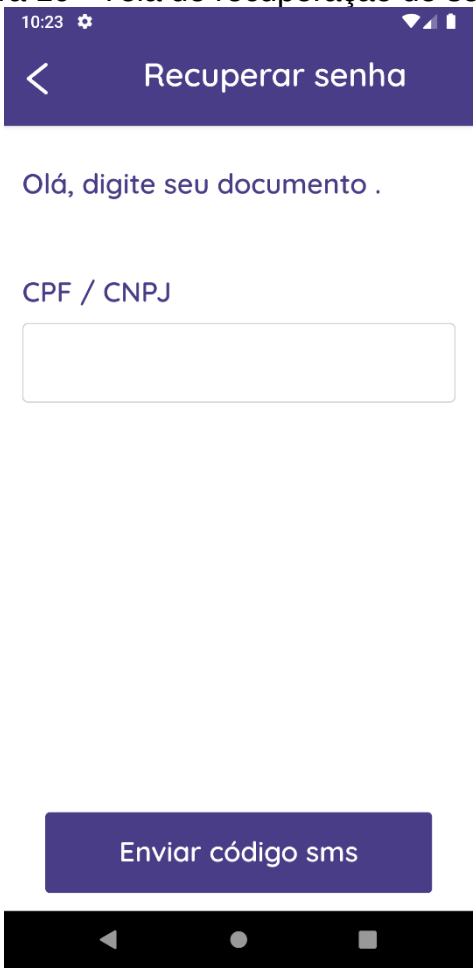
Figura 9 - Tela de login.

Fonte: Elaborado pelo autor.

5.3 Tela de Recuperação de Senha

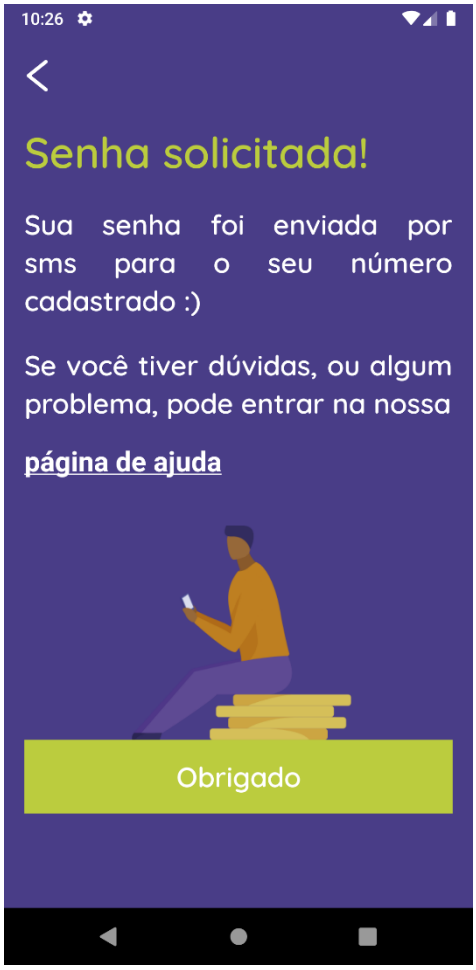
Nesta tela, o usuário deverá informar o CPF/CNPJ previamente cadastrado no aplicativo e, ao pressionar o botão “Enviar código sms”, o aplicativo irá validar se o CPF ou CNPJ corresponde a uma conta cadastrada. Caso a validação obtenha êxito, a nova tela com as instruções de troca de senha será enviada como visto na Figura 10.

Figura 10 - Tela de recuperação de senha.



Fonte: Elaborado pelo autor.

Figura 11 - Tela de envio de nova senha.

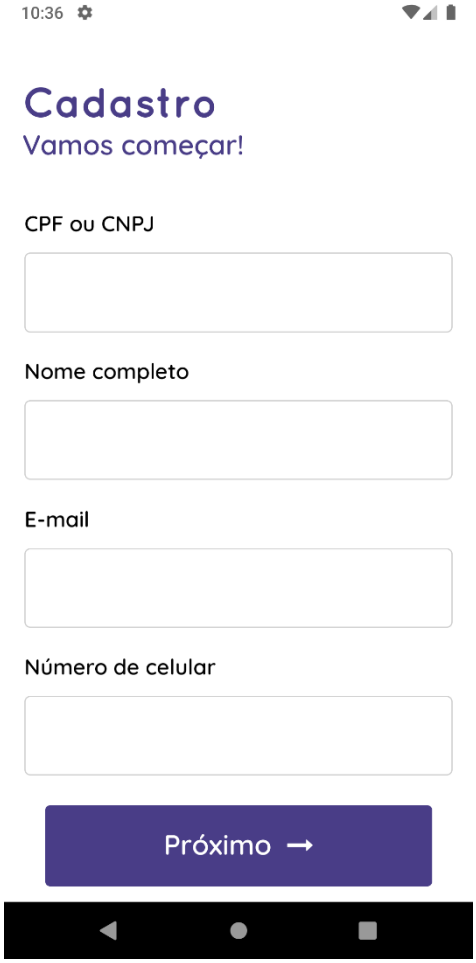


Fonte: Elaborado pelo autor.

5.4 Tela criar uma conta

Na tela inicial caso o usuário não tenha uma conta e quiser efetuar o cadastro, poderá clicar em "Criar conta", onde será redirecionado para uma tela de cadastro como mostra a figura 12. Será solicitado os dados cadastrais como CPF ou CNPJ, nome, telefone, endereço, foto do perfil e fotos dos documentos para comprovação. Após o envio dos dados será mostrado a tela que os dados foram enviados para análise.

Figura 12 - Tela novo cadastro.



Fonte: Elaborado pelo autor.

5.5 Tela principal

A figura 13 apresenta a tela principal do aplicativo após login, onde o usuário tem a opção de realizar as principais funções. Essa tela é composta pelos componentes:

- Transferir: Um botão para caso o usuário deseja efetuar transferência;
- Extrato: Um botão para caso o usuário deseja retirar um extrato das movimentações;
- Receber boleto ou PIX: Um botão para caso o usuário deseja gerar boletos ou QR Code PIX para receber pagamentos;
- Pagamentos: Um botão para caso o usuário deseja efetuar o pagamento de boletos.
- Ícone *Logoff*: Um botão para caso o usuário deseja sair da conta;

- Dados da conta: informações sobre agência, conta e nome são mostrados na tela, assim como o saldo disponível.

Figura 13 - Tela principal

Fonte: Elaborado pelo autor.

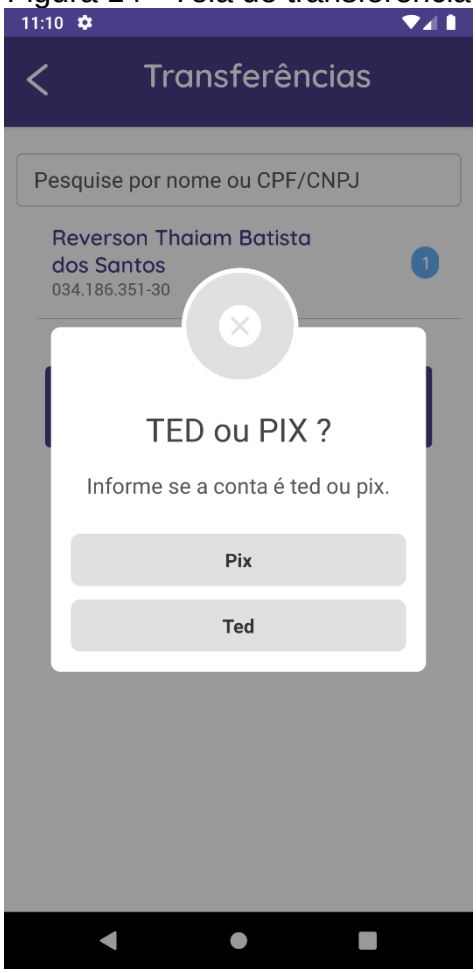
5.6 Tela de transferência

A figura 14 apresenta a tela de transferência do aplicativo, onde o usuário tem a opção de realizar transferência. Essa tela é composta pelos componentes:

- Pesquisar usuário já cadastrado anteriormente: Caso houver cadastro anteriormente, a lista de usuário é mostrada em tela;
- Pesquisar usuário: Um *input* para pesquisar por CPF ou CNPJ;

- Cadastrar pessoa: Caso o usuário não estiver cadastrado, ele poderá cadastrar a conta do usuário. Precisa selecionar se deseja cadastrar como Pix ou Ted como mostra a figura abaixo;

Figura 14 - Tela de transferência

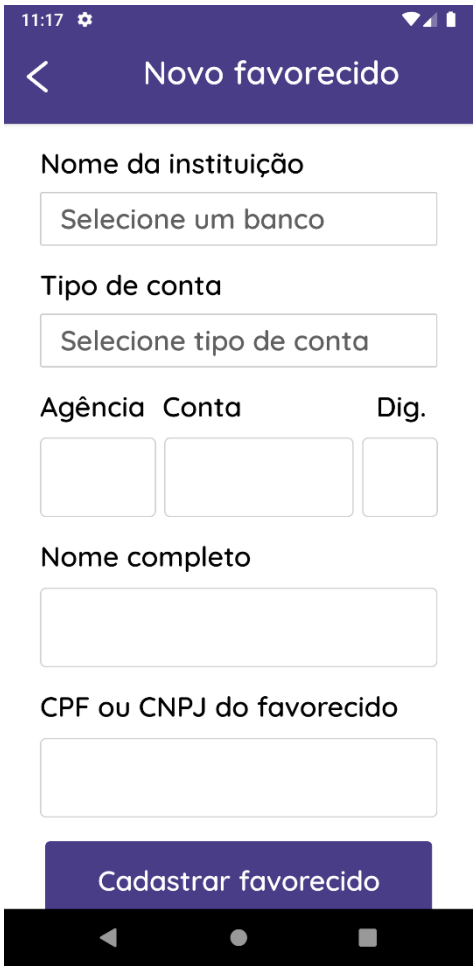


Fonte: Elaborado pelo autor.

- Pix: Caso o usuário selecionar Pix, precisa digitar o tipo de chave e nome do favorecido.
- TED: Caso o usuário selecionar TED, precisa digitar as informações da conta como:
    - Nome da instituição;
    - Tipo de conta;
    - Agência;
    - Conta;
    - Dígito;

- Nome completo;
- CPF ou CNPJ do favorecido.

Figura 15 - Tela da conta do favorecido.



Fonte: Elaborado pelo autor.

- Efetuar transferência: Será mostrado a informação da conta que irá ser favorecida para ser validada. O sistema mostrará a opção do valor a ser transferido, como mostra a figura abaixo. Após este processo o usuário poderá efetuar a transferência.

Figura 16 - Tela da validação dos dados.

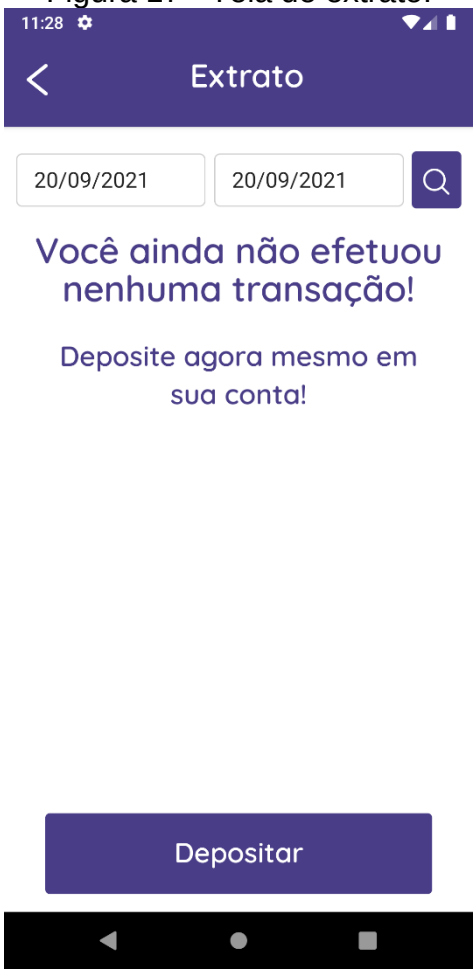
Fonte: Elaborado pelo autor.

5.7 Tela de extrato

A figura 17 apresenta a tela de extrato do aplicativo, onde o usuário tem a opção de visualizar o extrato de movimentações. Essa tela é composta pelos componentes:

- Pesquisar por um intervalo de data: Caso houver movimentações financeiras o sistema mostra como resultado de pesquisa no intervalo da data pré-selecionada;

Figura 17 - Tela de extrato.



Fonte: Elaborado pelo autor.

- Botão depositar: O sistema mostrará um botão "Depositar" caso o usuário não tenha nenhuma movimentação.

5.8 Tela recebimentos

A figura 18 apresenta a tela de recebimentos do aplicativo, onde o usuário tem a opção de gerar boletos ou QR Code PIX. Essa tela é composta pelos componentes:

- *Input* para digitar o valor a receber: O usuário deverá informar o valor para gerar o boleto ou Pix por QR Code;

Figura 18 - Tela de deposito.

Fonte: Elaborado pelo autor.

- Boleto: Caso a opção selecionada seja por boleto, mostrará a tela do código do boleto gerado para pagamento;

Figura 19 - Tela do código de barras.

Fonte: Elaborado pelo autor.

- Pix QR code: caso a opção selecionada seja por QR Code PIX, mostrará a tela do QR code para pagamento
    - Copiar código: ao copiar o código, é salvo na memória interna do celular podendo ser compartilhado;
    - Compartilhar: Poderá compartilhar o QR code por meios de transmissão.

Figura 20 - Tela do QR code PIX.

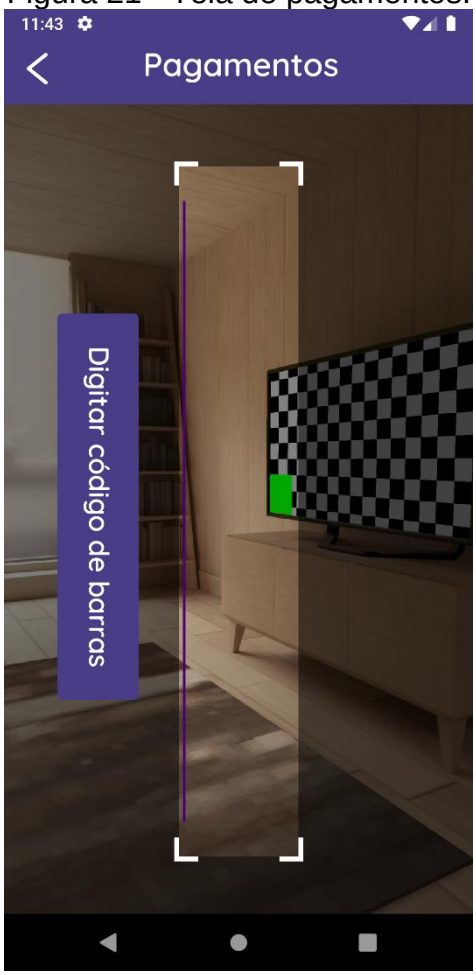
Fonte: Elaborado pelo autor.

5.9 Tela de pagamentos de boletos

A figura 21 apresenta a tela para efetuar pagamentos de boletos do aplicativo, onde o usuário tem a opção de fazer pagamentos por leitura de código de barra ou escrever o código. Essa tela é composta pelos componentes:

- Leitura do código usando a câmera: É aberto a câmera do celular para leitura do código de barras;

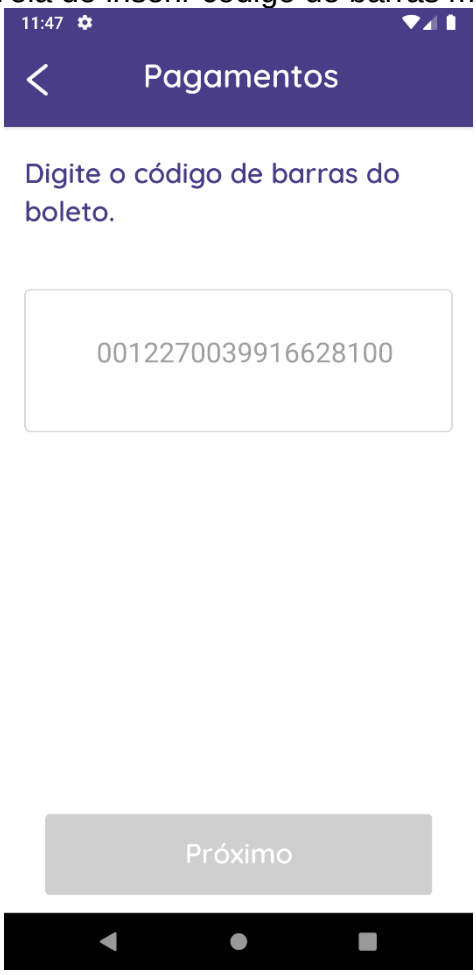
Figura 21 - Tela de pagamentos.



Fonte: Elaborado pelo autor.

- Botão para digitar: O sistema mostra o *input* para ser digitado o código ou colado pela memória interna.

Figura 22 - Tela de inserir código de barras manualmente.



Fonte: Elaborado pelo autor.

- Tela de confirmação: O sistema mostrará os dados do boleto inserido para validação. Após a confirmação o usuário, poderá confirmar o pagamento.

6 CONSIDERAÇÕES FINAIS

Este trabalho apresentou o desenvolvimento e a documentação do aplicativo voltado ao *front-end* de um banco digital, apresentando a sua justificativa, descrição geral, tecnologias utilizadas para o desenvolvimento e seus resultados.

Para a elucidação dos requisitos, foi realizada e documentada por meio de conhecimentos técnicos e teóricos sobre o funcionamento de um banco digital, e confeccionando o documento de *briefing*. Estes serviram de base para determinar as características do aplicativo e garantir com que os requisitos levantados fossem persistentes para efetuar todas as atividades operacionais básicas de um banco, com a utilização do *framework ReactJs*, linguagem de programação *Javascript.*

Através dos conceitos de UX e bibliotecas para determinadas funções no aplicativo, foi possível atender às necessidades do aplicativo, respeitando a experiência de usuário, como telas de fácil entendimento e objetivas. O desenvolvimento foi feito em tempo hábil e com sintaxe de fácil entendimento, mostrando a praticidade de uma linguagem moderna na construção de um aplicativo.

Os resultados obtidos foram satisfatórios, visto que o aplicativo mostrou a possibilidade de implementação de um problema que antes era considerado como muito difícil e com uma equipe maior de programadores, com todas as tecnologias atuais, fez que esse tipo de desenvolvimento se tornasse possível com menos recurso para ser implementado.

Além disso, a forma que foi construída deixa espaços para futuras implementações, com novos recursos para atender melhor o usuário, utilizando uma linguagem de programação moderna com vários recursos para serem utilizados, a fim de melhorar o desempenho, segurança e manutenção futura do aplicativo.

O código do aplicativo foi publicado no BitBucket e fica disponível para download como contribuição para trabalhos futuros.
https://ReversonThaiam@bitbucket.org/ReversonThaiam/banco-digital-tcc.git.

6.1 Trabalhos futuros

Este trabalho compõe apenas com a parte do *front-end da aplicação*, precisando ser integrado a um serviço de *back-end*, consumindo serviços utilizando os *endpoint* de terceiros, APIs e outros serviços ligados ao *back-end.*

Vale destacar que neste trabalho o layout pode ser aprimorado, utilizando os melhores conceitos de UX/UI. O design UX que é traduzido como "experiência de usuário" para trazer uma melhor usabilidade para o usuário que lhe atenda em todos os requisitos, trazendo satisfação ao utilizar a aplicação. E fazer uso do conceito de UI que em tradução literal é "interface de usuário ou *user interface*" que visa criar interfaces fáceis e amigáveis na visão do usuário.

Este trabalho pode ser complementado fazendo o banco de dados e regras de negócio do back-end.

Observando o cenário atual, o *back-end* poderia ser recomendado a utilização da linguagem NodeJS, para a construção das regras de negócio, tratamento dos dados e fazer as requisições com o banco de dados para inserir, ler e arquivar informações. Utilizando o MVC como referência o NodeJS poderia ser utilizado na camada de *controller*.

Apesar dos resultados mostrados serem satisfatórios, este trabalho poderia alcançar um nível de significância maior atendendo esses requisitos levantados anteriormente. São pontos que podem ser explorados em trabalhos futuros.

REFERÊNCIAS

1. BANCO CENTRAL DO BRASIL. PROER: programa estimula a reestruturação de bancos. Brasília, DF: BACEN, 2019. Disponível em: <https://www.bcb.gov.br/htms/proer.asp?frame=1>. Acesso em: 03 nov. 2021
2. ITU. Youth are at forefront of internet adoption. ITU, 2017. Disponível em: <https://www.itu.int/en/ITUD/Statistics/Documents/facts/ICTFactsFigures2017.pdf> Acesso em: 28 abr. 2021.
3. ENGEL, J. F.; BLACKWELL, R. D.; MINIARD, P. W. Comportamento do consumidor. 8. ed. Rio de Janeiro: LTC, 2000.
4. TUTORIAL. Tutorial: **Intro to React**, 2019. Disponível em: < https://reactjs.org/tutorial/tutorial.html#what-is-react>. Acesso em: 05 mai. 2020.
5. OCCHINO, Tom. **React Native:** Bringing modern web techniques to mobile, 2015. Disponível em: <https://engineering.fb.com/2015/03/26/android/react-native-bringing-modern-web-techniques-to-mobile/>. Acesso em: 08 setembro. 2021.

   FLANAGAN, David. **Javascript o guia definitivo**. 6° Ed. Porto Alegre. Bookman editora. 2013.
6. CERQUEIRA, Jessica S.; CORREIA, Saulo P.; FERREIRA, Alex S.; ROBSON, Hebraico C. **Automação para fiscalização de CNH utilizando dispositivos móveis e biometria.** Contribuições, 1 Faculdade de Tecnologia e Ciências (FTC), 2 Universidade Estadual do Sudoeste da Bahia (UESB), 3 Universidade Federal do Recôncavo da Bahia (UFRB). ISSN 2178-0471. Vol. 1 n. 7 Mar. 2016. pág. 1-13.
7. BORGES, Frank. **BANCOS DIGITAIS X BANCOS TRADICIONAIS:** UMA ANÁLISE DAS IMPLICAÇÕES CAUSADAS PELOS BANCOS DIGITAIS NO MERCADO BANCÁRIO BRASILEIRO. Uberlândia. 2019. Disponível em: < https://repositorio.ufu.br/bitstream/123456789/28298/7/BancosDigitaisTradicionais.pdf>. Acesso em: 03 nov. 2021.
8. MOZILLA.ORG. JavaScript. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Glossary/JavaScript>. Acesso: em 21 nov. 2021.

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
GABINETE DO REITOR

Av. Universitária, 1069 ● Setor Universitário
Caixa Postal 86 ● CEP 74605-010
Goiânia ● Goiás ● Brasil
Fone: (62) 3946.1000
www.pucgoias.edu.br ● reitoria@pucgoias.edu.br

RESOLUÇÃO n° 038/2020 – CEPE

ANEXO I

APÊNDICE ao TCC

Termo de autorização de publicação de produção acadêmica

O(A) estudante Reverson Thaiam Batista dos Santos
do Curso de Ciência da Computação, matrícula 2015.2.0028.0133.5,
telefone: (62) 994330465 e-mail reversonthayan@gmail.com, na qualidade de titular dos direitos autorais, em consonância com a Lei nº 9.610/98 (Lei dos Direitos do autor), autoriza a Pontifícia Universidade Católica de Goiás (PUC Goiás) a disponibilizar o Trabalho de Conclusão de Curso intitulado Desenvolvimento de um aplicativo mobile para banco digital, gratuitamente, sem ressarcimento dos direitos autorais, por 5 (cinco) anos, conforme permissões do documento, em meio eletrônico, na rede mundial de computadores, no formato especificado (Texto (PDF); Imagem (GIF ou JPEG); Som (WAVE, MPEG, AIFF, SND); Vídeo (MPEG, MWV, AVI, QT); outros, específicos da área; para fins de leitura e/ou impressão pela internet, a título de divulgação da produção científica gerada nos cursos de graduação da PUC Goiás.

Goiânia, 01 de dezembro de 2021.

Assinatura do(s) autor(es): 

Nome completo do autor: Reverson Thaiam Batista dos Santos

Assinatura do professor-orientador: 

Nome completo do professor-orientador: Lucília Gomes Ribeiro