PROJETO “BANCO ACE” EM JAVA

Bruno Sérgio da Costa[1]
José André da Costa Neto[2]
Luan das Neves Monteiro Cardoso[3]
Pedro Henrique Leal Vieira[4]
Messias Rafael Batista[5]

RESUMO

Este resumo apresenta um sistema financeiro desenvolvido por meio da implementação de classes em linguagem Java. O objetivo do projeto foi criar um software capaz de oferecer funcionalidades relacionadas a investimentos, pagamentos, empréstimos e gerenciamento de conta poupança, com o intuito de proporcionar aos usuários maior controle e facilidade na gestão de suas finanças pessoais. A metodologia adotada consistiu na criação de classes específicas para cada funcionalidade do sistema, permitindo uma estrutura modular e expansível. Destacam-se as classes "Investimentos", responsável pela realização e consulta de investimentos, "Pagamentos", que possibilita efetuar pagamentos diversos, e "Conta Poupanca", responsável pelo gerenciamento da conta poupança do usuário. Os resultados obtidos demonstraram um sistema funcional e eficiente, capaz de atender às necessidades dos usuários. As funcionalidades implementadas permitiram a realização de investimentos com cálculo de rentabilidade, a efetuação de pagamentos de diferentes tipos, como cartão de crédito, boletos e empréstimos, além do gerenciamento da conta poupança com adição de valores e cálculo de rendimento. O sistema desenvolvido apresentou-se intuitivo e de fácil utilização, proporcionando aos usuários uma experiência agradável na gestão de suas finanças pessoais. A modularidade do código permitiu a manutenção e expansão do sistema, possibilitando a inclusão de novas funcionalidades no futuro. Em suma, o sistema financeiro desenvolvido mostrou-se uma solução eficaz para o controle financeiro pessoal, oferecendo aos usuários uma ferramenta prática e versátil para a gestão de suas finanças.

Palavras- chave: Banco digital; Java; Código.

[1] Graduando do Curso de Sistemas de Informação E-mail: 2022211790019@iesp.edu.br
[2] Graduando do Curso de Sistemas de Informação E-mail: 2023111510072@iesp.edu.br
[3] Graduando do Curso de Sistemas de Informação E-mail: 2022210220009@iesp.edu.br
[4] Graduando do Curso de Sistemas de Informação E-mail: 2022210220031@iesp.edu.br
[5] Professor Orientador. Docente do Curso Superior em Sistemas de Informação. E-mail: prof2118@iesp.edu.br

ABSTRACT

This abstract presents a financial system developed through the implementation of classes in the Java programming language. The objective of the project was to create a software capable of providing functionalities related to investments, payments, loans, and savings account management, aiming to offer users greater control and ease in managing their personal finances. The methodology adopted involved creating specific classes for each functionality of the system, allowing for a modular and expandable structure. Notable classes include "Investimentos," responsible for making and consulting investments, "Pagamentos" which enables various types of payments such as credit card, bills, and loans, and "ContaPoupanca," responsible for managing the user's savings account. The obtained results demonstrated a functional and efficient system that meets users' needs. The implemented functionalities allowed for investment operations with profitability calculations, the execution of different types of payments, and the management of the savings account, including value addition and interest calculation. The developed system proved to be intuitive and user-friendly, providing users with a pleasant experience in managing their personal finances. The code's modularity allowed for system maintenance and expansion, enabling the inclusion of new functionalities in the future. In summary, the developed financial system proved to be an effective solution for personal financial control, offering users a practical and versatile tool for managing their finances.

Keywords: Digital bank; Java; Code

1 INTRODUÇÃO

A fim de atender a demanda tecnológica e social, o projeto se dispôs a conceder ao público não apenas um produto genérico que tanto se manifesta no mercado atual, mas sim uma experiência que promete, e de fato cumpre, ser inovadora, simples e principalmente acessível, visando não só o público geral, mas todos os nichos da população, visto que este representa um dos, se não o maior, problema enfrentado pelas grandes companhias que visam sua expansão e sucesso.

Neste documento, apresentamos o desenvolvimento de um programa na linguagem Java, este que consiste em um banco digital com diversas funcionalidades. A finalidade deste projeto é simular ao usuário uma experiência bancária mais acessível e prática, sem a necessidade de deslocamentos físicos a uma agência bancária. A justificativa para o desenvolvimento deste programa se baseia na demanda cada vez maior de serviços bancários online de excelência, bem como na necessidade de oferecer ao usuário uma interface intuitiva e fácil de utilizar, principalmente que leve as demandas de acessibilidade como prioridade.

O objetivo principal é desenvolver um programa que ofereça ao usuário diversas funcionalidades, como: conta corrente, cartão de crédito, empréstimo e investimento. Com base nessa visão geral do produto, iniciamos a etapa de elicitação de requisitos,

observando as principais funções presentes nos principais aplicativos bancários do mercado. Dessa forma, identificamos as funcionalidades que seriam essenciais para o desenvolvimento do nosso banco digital, como a abertura de conta corrente, emissão de cartão de crédito, solicitação de empréstimo e investimento. A partir dessas funcionalidades, criamos as classes, métodos e atributos necessários para implementá-las de forma eficiente e segura.

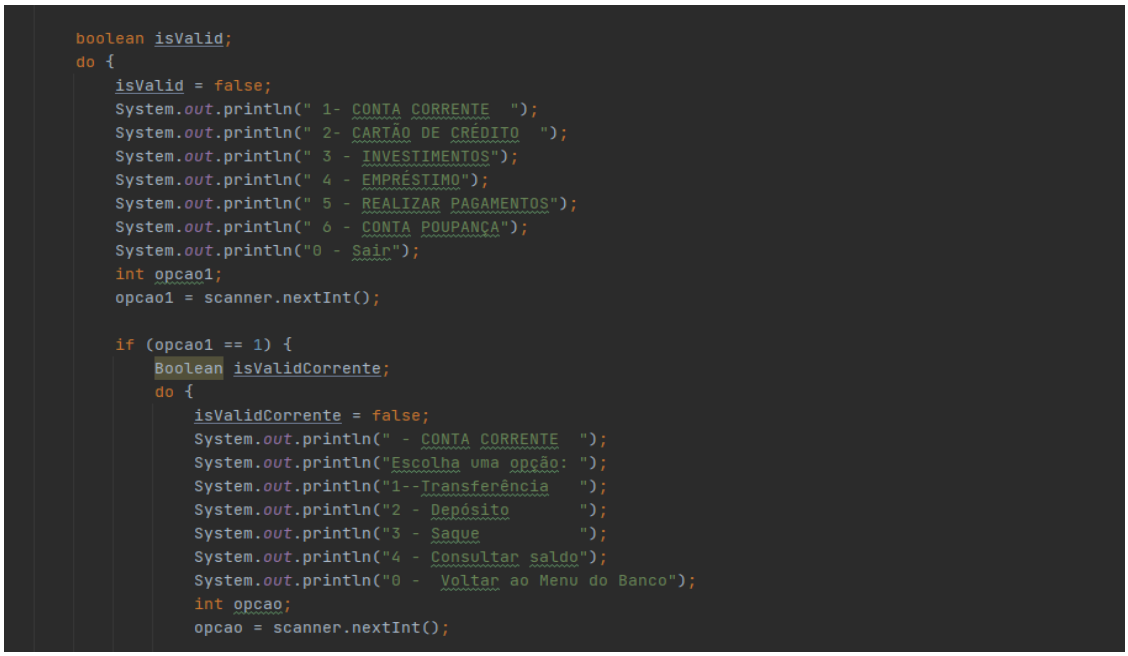
2 FUNDAMENTAÇÃO TEÓRICA

A utilização de Ambientes de Desenvolvimento Integrado (IDEs) e sistemas de controle de versão desempenharam um papel fundamental no processo de desenvolvimento do software. Neste estudo de caso, exploraremos a combinação do IntelliJ como IDE e o GitHub como plataforma de controle de versão para o desenvolvimento de um projeto em Java.

O IntelliJ foi escolhido como a IDE para este projeto devido às suas amplas funcionalidades e facilidade de uso. O IntelliJ fornece um conjunto de recursos que auxiliam os desenvolvedores, como realce de sintaxe, depuração integrada, gerenciamento de dependências e uma interface intuitiva. Além disso, sua integração com o sistema de controle de versão facilita o processo de colaboração em equipe.

Como plataforma para controle de versão, foi utilizado o GitHub. Por meio dele, é possível gerenciar alterações no código-fonte, rastrear problemas, colaborar com outros desenvolvedores e disponibilizar o projeto para a comunidade. O GitHub permite o uso de branches, facilitando o trabalho em paralelo, e também possui recursos para revisão de código, tornando o processo mais eficiente e confiável.

Durante o desenvolvimento do , foram realizadas diversas modificações utilizando o controle de versão do GitHub. Foram feitos envios (commits) contendo pequenas mudanças, como melhorias na interface da classe principal, que envolve o sistema de menus e submenus, aprimorando a experiência do usuário. Além disso, novas classes foram adicionadas ao projeto, como a classe "Conta Poupança", para a implementação de funcionalidades específicas desse tipo de conta.



```
boolean isValid;
do {
    isValid = false;
    System.out.println(" 1- CONTA CORRENTE  ");
    System.out.println(" 2- CARTÃO DE CRÉDITO  ");
    System.out.println(" 3 - INVESTIMENTOS");
    System.out.println(" 4 - EMPRÉSTIMO");
    System.out.println(" 5 - REALIZAR PAGAMENTOS");
    System.out.println(" 6 - CONTA POUPANÇA");
    System.out.println("0 - Sair");
    int opcao1;
    opcao1 = scanner.nextInt();

    if (opcao1 == 1) {
        Boolean isValidCorrente;
        do {
            isValidCorrente = false;
            System.out.println(" - CONTA CORRENTE  ");
            System.out.println("Escolha uma opção: ");
            System.out.println("1--Transferência   ");
            System.out.println("2 - Depósito       ");
            System.out.println("3 - Saque          ");
            System.out.println("4 - Consultar saldo");
            System.out.println("0 -  Voltar ao Menu do Banco");
            int opcao;
            opcao = scanner.nextInt();
```

Exemplo do sistema de menus e submenus no código.

A utilização do IntelliJ como IDE proporcionou um ambiente de desenvolvimento produtivo e eficiente, permitindo a escrita de código mais ágil e a identificação rápida de erros por meio de recursos de depuração integrados. Já o controle de versão pelo GitHub trouxe benefícios como o histórico completo de alterações, a possibilidade de rastrear problemas e a colaboração em equipe de forma simplificada. O uso de um Ambiente de Desenvolvimento Integrado (IDE) como o IntelliJ e um sistema de controle de versão como o GitHub foram essenciais para facilitar o processo de desenvolvimento.

3 METODOLOGIA

No código em questão, a classe principal, denominada Principal, desempenha um papel central no sistema financeiro. Essa classe é responsável por apresentar um menu de opções ao usuário e coordenar a interação com as outras classes do sistema.

A classe Principal atua como o ponto de partida do sistema, fornecendo uma interface interativa por meio da qual o usuário pode acessar as diferentes funcionalidades financeiras disponíveis. Ao ser executada, a classe exibe um menu com opções, permitindo que o usuário escolha entre consultar rentabilidade de investimentos, realizar pagamentos, gerenciar empréstimos e interagir com a conta de poupança.

Ao receber a escolha do usuário, a classe Principal instancia objetos das classes apropriadas, como Investimentos, Pagamentos, Empréstimo e ContaPoupanca, e chama os métodos correspondentes para executar as operações desejadas. Essa classe atua como uma espécie de controlador, coordenando a interação entre o usuário e as funcionalidades do sistema.

Através da classe Principal, o usuário pode interagir com o sistema de forma intuitiva e realizar diversas operações financeiras. As outras classes, como Investimentos, Pagamentos, Emprestimo e ContaPoupanca, são responsáveis por implementar as funcionalidades específicas de cada área financeira, enquanto a classe Principal atua como uma camada de abstração para facilitar o uso do sistema.

Em resumo, a classe Principal desempenha um papel fundamental no sistema financeiro, fornecendo uma interface amigável e coordenando as interações entre o usuário e as funcionalidades disponíveis. Ela é responsável por apresentar um menu de opções, receber as entradas do usuário, invocar os métodos apropriados das outras classes e exibir as informações relevantes de volta ao usuário, proporcionando uma experiência interativa e funcional para gerenciar as finanças.

4 RESULTADO E DISCUSSÃO

O objetivo do projeto foi criar um software capaz de realizar operações financeiras, como investimentos, pagamentos, empréstimos e gerenciamento de conta poupança, proporcionando aos usuários maior controle e facilidade na gestão de suas finanças pessoais.

O produto resultante desse projeto consiste em um sistema financeiro implementado em linguagem Java, composto por várias classes que interagem entre si. O código foi estruturado de forma modular, permitindo uma fácil manutenção e expansão das funcionalidades.

Uma das classes principais do sistema é a "Principal", responsável por apresentar um menu de opções ao usuário e direcionar as operações para as respectivas classes correspondentes. Dessa forma, o usuário pode realizar diversas operações financeiras de maneira intuitiva e conveniente.

Destacam-se algumas funcionalidades importantes implementadas no código:

- Classe Investimentos: Essa classe permite ao usuário realizar investimentos e consultar a rentabilidade estimada. O método realizar Investimento realiza o investimento ao longo de um determinado período, acumulando valor investido e rendimentos. Já o método consultar Rentabilidade permite ao usuário verificar o rendimento estimado com base no tempo de investimento.
- Classe Pagamentos: Essa classe oferece a funcionalidade de efetuar pagamentos, como pagamentos de cartão de crédito, boletos e empréstimos. O método efetuar Pagamento exibe um menu de opções ao usuário, permitindo que ele escolha o tipo de pagamento desejado. Em seguida, o sistema solicita os valores necessários e realiza o pagamento de acordo com as regras estabelecidas.
- Classe Conta Poupança: Essa classe gerencia a conta poupança do usuário. Ela permite adicionar valores à conta e consultar o saldo atual. Além disso, possui o método rendimento Poupança, responsável por aplicar o rendimento configurado sobre o saldo da conta poupança.

O sistema financeiro desenvolvido atende aos requisitos propostos, fornecendo uma solução eficiente e funcional para a gestão das finanças pessoais. Com a modularidade do código, o sistema pode ser facilmente adaptado e expandido para incluir novas funcionalidades no futuro.

Durante o desenvolvimento, alguns desafios foram enfrentados, como garantir a correta interação entre as classes e o tratamento adequado de entradas do usuário. No entanto, esses desafios foram superados com a aplicação de boas práticas de programação e o uso de estruturas de controle adequadas.

Uma das principais vantagens do sistema é sua facilidade de uso, permitindo que mesmo usuários sem conhecimento técnico avançado possam realizar operações financeiras de forma intuitiva. Além disso, a modularidade do código facilita a manutenção e futuras melhorias do sistema.

O projeto de desenvolvimento do sistema financeiro interativo resultou em um produto funcional e eficiente, capaz de auxiliar os usuários na gestão de suas finanças pessoais. As funcionalidades implementadas nas classes Investimentos

4.1 Diagrama de Caso de Uso

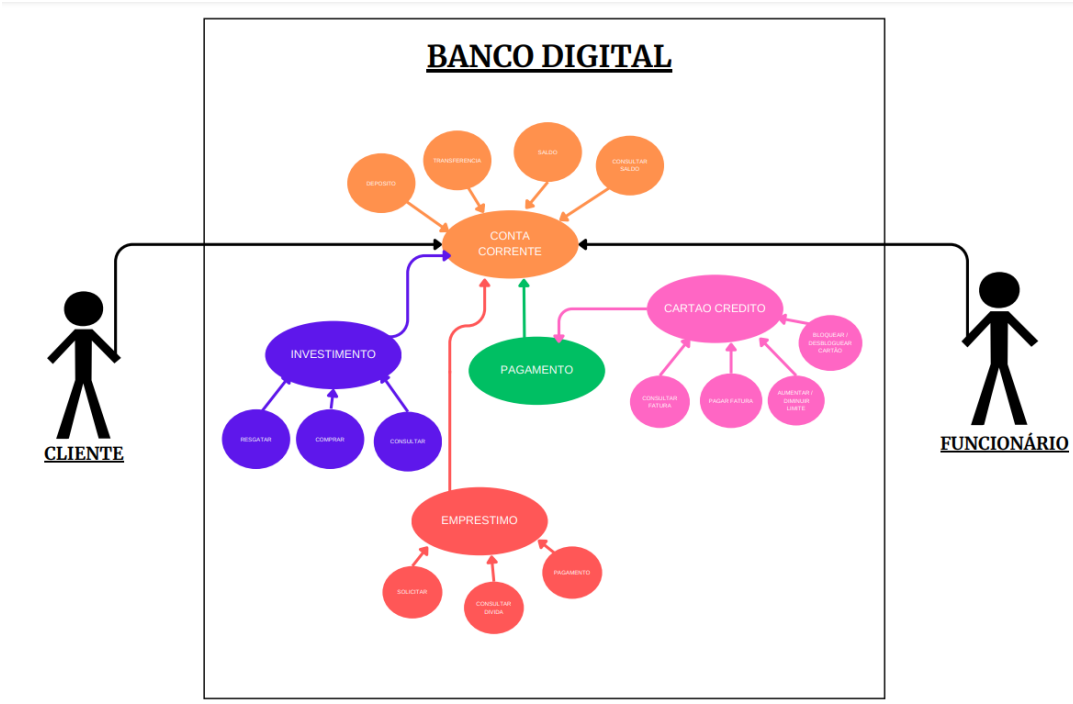


Figura 1 - Diagrama de Caso de Uso

Disponível em: https://github.com/PedroLeal1/BancoDigitalUniesp/blob/master/Diagrama%20de%20Caso%20Uso.pdf

Em conjunto, esses casos de uso fornecem um panorama abrangente das principais funcionalidades do sistema financeiro, abordando aspectos como investimentos, pagamentos, gerenciamento de conta poupança e monitoramento de dívidas. O diagrama de caso de uso facilita a compreensão dessas interações e destaca a maneira como o sistema atende às necessidades dos usuários, auxiliando-os no controle de suas finanças pessoais.

4.2 Diagrama de Classe

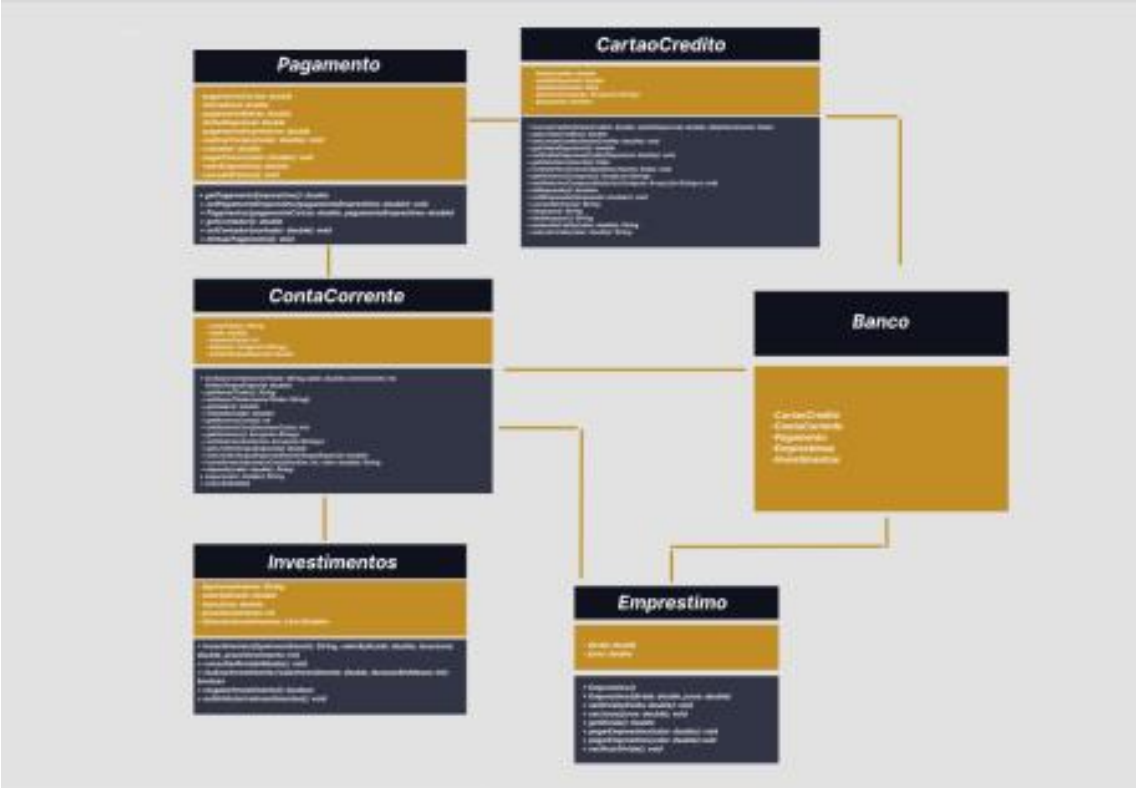


Figura 2 - Diagrama de Classe
Disponível em: https://github.com/PedroLeal1/BancoDigitalUniesp/blob/master/Diagrama%20de%20Classes.pdf

A figura representa as classes do projeto e as relações diretas que as respectivas possuem, destaca-se que a classe BANCO representa uma espécie de "ponto de partida" para todo o fluxo natural do projeto.

5 CONSIDERAÇÕES FINAIS

Durante o desenvolvimento do projeto, foram obtidos resultados significativos que proporcionaram uma resposta efetiva ao problema inicial apresentado. O objetivo de criar um sistema financeiro capaz de oferecer funcionalidades relacionadas a investimentos, pagamentos, empréstimos e gerenciamento de conta poupança foi alcançado com sucesso.

A implementação das diversas classes, como "Investimentos", "Pagamentos", "Empréstimo" e "ContaPoupanca", permitiu a criação de um software funcional e de

fácil utilização. Os usuários foram capazes de realizar investimentos com cálculo de rentabilidade, efetuar pagamentos de diferentes tipos, gerenciar empréstimos e controlar sua conta poupança de forma prática e intuitiva.

Ao longo do desenvolvimento, destacaram-se trechos de código importantes que contribuíram para a solução do problema. A classe "Investimentos" possibilitou a realização de operações financeiras, considerando taxas de juros e prazos, enquanto a classe "Pagamentos" permitiu o processamento de diferentes tipos de pagamentos de forma segura e confiável. A classe "Emprestimo" auxiliou no gerenciamento das dívidas e na atualização dos valores devidos, e a classe "ContaPoupanca" facilitou o controle e cálculo do rendimento da conta.

Esses resultados evidenciam a eficácia do projeto em fornecer uma solução abrangente e funcional para a gestão financeira pessoal. O sistema desenvolvido contribui para que os usuários tenham maior controle sobre suas finanças, facilitando o acompanhamento de investimentos, pagamentos e empréstimos, além de proporcionar uma maneira eficiente de gerenciar a conta poupança.

Portanto, o projeto obteve sucesso ao solucionar o problema inicial e proporcionar aos usuários uma ferramenta efetiva e confiável para o gerenciamento de suas finanças pessoais.

5.1 AMEAÇAS AO PROJETO

Durante o desenvolvimento do projeto, algumas dificuldades podem surgir e ameaçar a validade do projeto. Algumas ameaças possíveis são:

Limitações de funcionalidades: O projeto atual aborda funcionalidades básicas de um sistema financeiro, mas pode não atender a todas as necessidades e requisitos específicos de diferentes usuários. Isso pode limitar sua utilidade e adoção em cenários mais complexos.

Falhas de segurança: O código atual não aborda medidas de segurança robustas, como criptografia de dados, autenticação forte ou proteção contra ataques cibernéticos. Essas falhas podem tornar o sistema vulnerável a violações de segurança e comprometer a integridade das informações financeiras dos usuários.

5.2 TRABALHOS FUTUROS

Existem diversas melhorias e funcionalidades adicionais que podem ser implementadas para aprimorar o projeto e melhorar sua performance. Algumas sugestões de trabalhos futuros incluem:

Refinamento da interface do usuário: Uma interface mais intuitiva e amigável pode melhorar a experiência do usuário ao interagir com o sistema financeiro. Isso pode envolver a implementação de gráficos, recursos visuais e uma navegação mais fluida.

Adição de recursos avançados: Recursos adicionais, como planejamento financeiro, cálculos de juros compostos, notificações de vencimentos e orçamentação, podem fornecer mais controle e insights sobre as finanças dos usuários.

Implementação de segurança robusta: Introduzir medidas de segurança robustas, como autenticação de dois fatores, criptografia de dados sensíveis e monitoramento de atividades suspeitas, ajudará a proteger as informações financeiras dos usuários e garantir a confidencialidade e integridade dos dados.

Integração com serviços bancários: Integrar o sistema com APIs de instituições financeiras pode permitir aos usuários acessar informações de contas bancárias, realizar transferências eletrônicas, automatizar pagamentos e obter um panorama financeiro mais abrangente.

Testes e otimização: Realizar testes abrangentes, como testes de unidade, integração e desempenho, é fundamental para garantir a estabilidade e eficiência do sistema. Identificar possíveis problemas de desempenho e otimizar o código pode melhorar a velocidade e a capacidade de resposta do sistema.

Suporte a múltiplas moedas: Adicionar suporte a diferentes moedas e taxas de câmbio pode ser útil para usuários que lidam com transações internacionais e desejam ter uma visão abrangente de suas finanças em diferentes moedas.

Essas são apenas algumas sugestões de melhorias e funcionalidades futuras. O projeto pode ser expandido e aprimorado de acordo com as necessidades e requisitos específicos, visando sempre proporcionar uma experiência financeira mais completa e eficiente aos usuários

REFERÊNCIAS

ORACLE. Java SE Documentation. Disponível em: https://docs.oracle.com/javase/. Acesso em: 05/05/2023

UDEMY. Fundamentos de Programação com Java. Disponível em: https://www.udemy.com/course/fundamentos-de-programacao-com-java/. Acesso em: 08/05/2023