BancoAluno: DESENVOLVIMENTO DE UM SISTEMA BANCÁRIO PARA O PROJETO INTEGRADOR

Gabriel Aguiar do Nascimento[1]

RESUMO (Unidade 2)

Projeto de faculdade que visa a criação de um sistema bancário com funcionalidades básicas e intuitivas, com o objetivo de fornecer uma plataforma simples para as operações bancárias. Têm como principais funcionalidades a abertura de novas contas, tipos de contas variados (Conta corrente, Conta investimento, Conta poupança, entre outras) e funcionalidades padrões de um sistema bancário (Deposito, Saque, Transferência e Consulta de saldo).

A metodologia do projeto foi a implementação das funcionalidades utilizando a linguagem de programação Java, a modelagem do projeto com os Diagrama de casos de uso e Diagrama de classes e seguiu os requisitos que foram apresentados em sala de aula pelo professor.

Os resultados obtidos mostraram eficiência e viabilidade do sistema bancário. Por mais que seja um projeto inicial que apenas é usado pelo prompt de comando e não está conectado a nenhum banco de dados e não possui front-end, porém ele é bastante intuitivo e de fácil utilização.

Palavras-chaves: Sistema Bancário; Java; Funcionalidades Básicas; Intuitivo.

ABSTRACT (Unidade 2)

College project aimed at creating a banking system with basic and intuitive functionalities, with the goal of providing a simple platform for banking operations. Its main features include opening new accounts, various types of accounts (Checking Account, Investment Account, Savings Account, among others), and standard banking functionalities (Deposit, Withdrawal, Transfer, and Balance Inquiry).

The project methodology involved implementing the functionalities using the Java programming language, modeling the project with Use Case Diagrams and Class Diagrams, and adhering to the requirements presented by the professor in the classroom.

The obtained results demonstrated the efficiency and viability of the banking system. Although it is an initial project that is only used through the command prompt and is not connected to any database nor has a front-end, it is highly intuitive and easy to use.

Keywords: Banking System; Java; Basic Functionalities; Intuitive.

1 INTRODUÇÃO (Unidade 1)

O projeto tem como finalidade o desenvolvimento de um aplicativo para Bancos Digitais, visando criar um sistema bancário com diversas opções para os usuários, utilizando conceitos de POO (Programação Orientada a Objetos), para melhor organização do projeto foi utilizado também alguns conhecimentos em Diagrama de casos de uso e Diagrama de classes. O objetivo do projeto visa o desenvolvimento e manipulação de classes, como também aprimorar os conhecimentos de conceitos na linguagem de programação Java.

O desenvolvimento desse projeto é de importância para a disciplina de Introdução a Programação, pois utiliza-se do conceito de Programação Orientada a Objetos que atualmente, a maioria das linguagens de programação, necessitam que o programador tenha ao menos conhecimento básico sobre esse assunto.

2 FUNDAMENTAÇÃO TEÓRICA (Unidade 2)

O projeto foi desenvolvido na linguagem de programação Java(https://www.java.com/pt-BR/), utilizando em muitos casos os conceitos de Programação Orientada a Objetos (POO), sendo eles, encapsulamento, composição, herança, polimorfismo, classes e métodos abstratos e métodos construtores. Como no projeto foi de cunho educativo e tendo em vista que seria apenas uma aplicação utilizando apenas prompt de comando, apenas o aplicativo da IDE foi necessária para fazer e rodar o código do “BancoAluno”. A IDE utilizada para compilar o código foi a IntelliJ (https://www.jetbrains.com/pt-br/idea/).

A parte organizacional do projeto foi feito utilizando o Diagrama de casos de uso(4.1) e o Diagrama de classes(4.2), todos os dois diagramas foram feitos pela web no site do draw.io (https://www.drawio.com/).

2.2.1 **Este é um sub-tópico, caso precise**

3 METODOLOGIA (Unidade 2)

A primeira ferramenta a ser utilizada foi a site draw.io com o intuito de criar o Diagrama de casos de uso e o Diagrama de classes, esses dois diagramas serviram para uma melhor organização do projeto e para dar uma direção sobre o que deveria ser codificado e a ordem da codificação.

Em seguida o aplicativo Intellij foi utilizado para fazer o código do projeto e rodar o projeto no terminal do aplicativo, nele também foram feitos os testes para otimização do código.

O código do projeto ficará armazenado no seu respectivo repositório do github, lugar onde será postada novas atualizações para o melhoramento do programa.

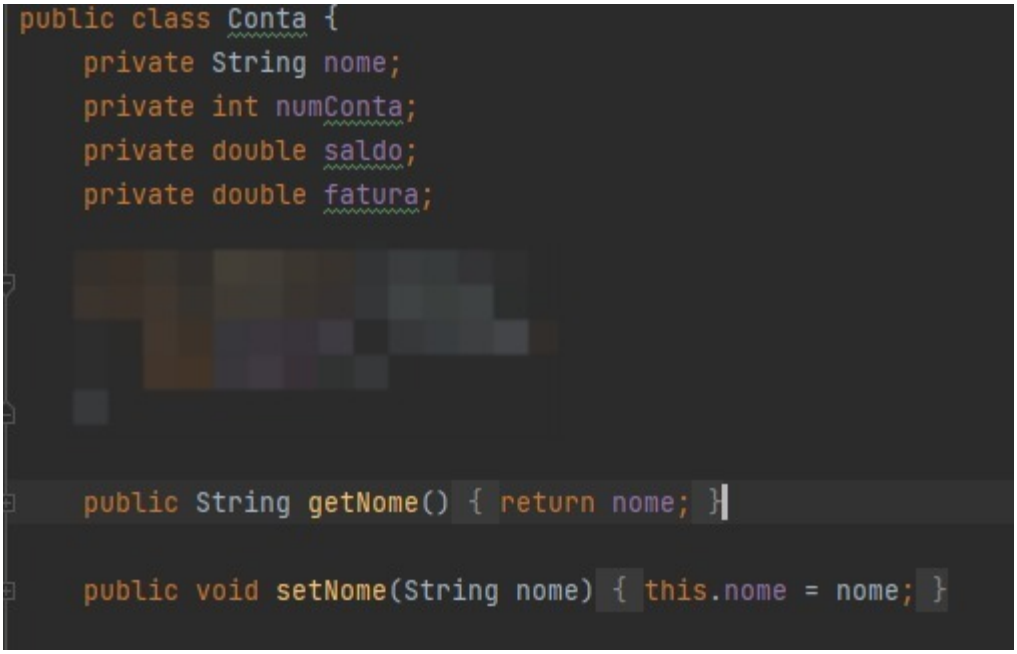
3.1 Projeto de leitura para remição do apenado

4 RESULTADO E DISCUSSÃO (Unidade 2)

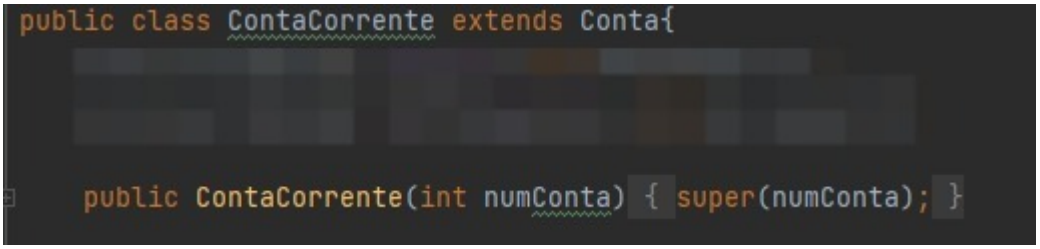
O projeto foi baseado na Programação Orientada a Objetos, respeitando as normas de encapsulamento, composição, herança, polimorfismo, classes e métodos abstratos e construtores.

O encapsulamento foi feito nos atributos que se referem a conta, privando esses atributos, o programa só poderá chamar ou alterar os atributos se utilizarem os métodos específicos para tal ação, como exemplo, o "get<nomeAtributo>()" para exibir algum conteúdo, e o ”set<nomeAtributo>()” para alterar o atributo.

Exemplo do atributo nome na classe Conta.



Depois do encapsulamento, o sistema de composição e de herança irá fazer com que o sistema faça relacionamentos entre classes e objetos. A herança irá derivar uma classe de outra, utilizando a palavra reservada “extends”. o exemplo a seguir mostra que a classe ContaCorrente se extende a classe Conta, e no construtor da classe ContaCorrente utilizará um atributo super que irá trazer o atributo “numConta” da classe Conta.



Os métodos construtores estão presentes em todas as classes do código. A única diferença é que na classe Conta o atributo “numConta” será o atributo que estará presente em todos os outros construtores que servirá para simbolizar que as outras classes de alguma forma se extendem a classe Conta

```
public class Conta {

    public Conta(int numConta) {
        this.numConta = numConta;
        this.saldo = 0;
    }
```

.

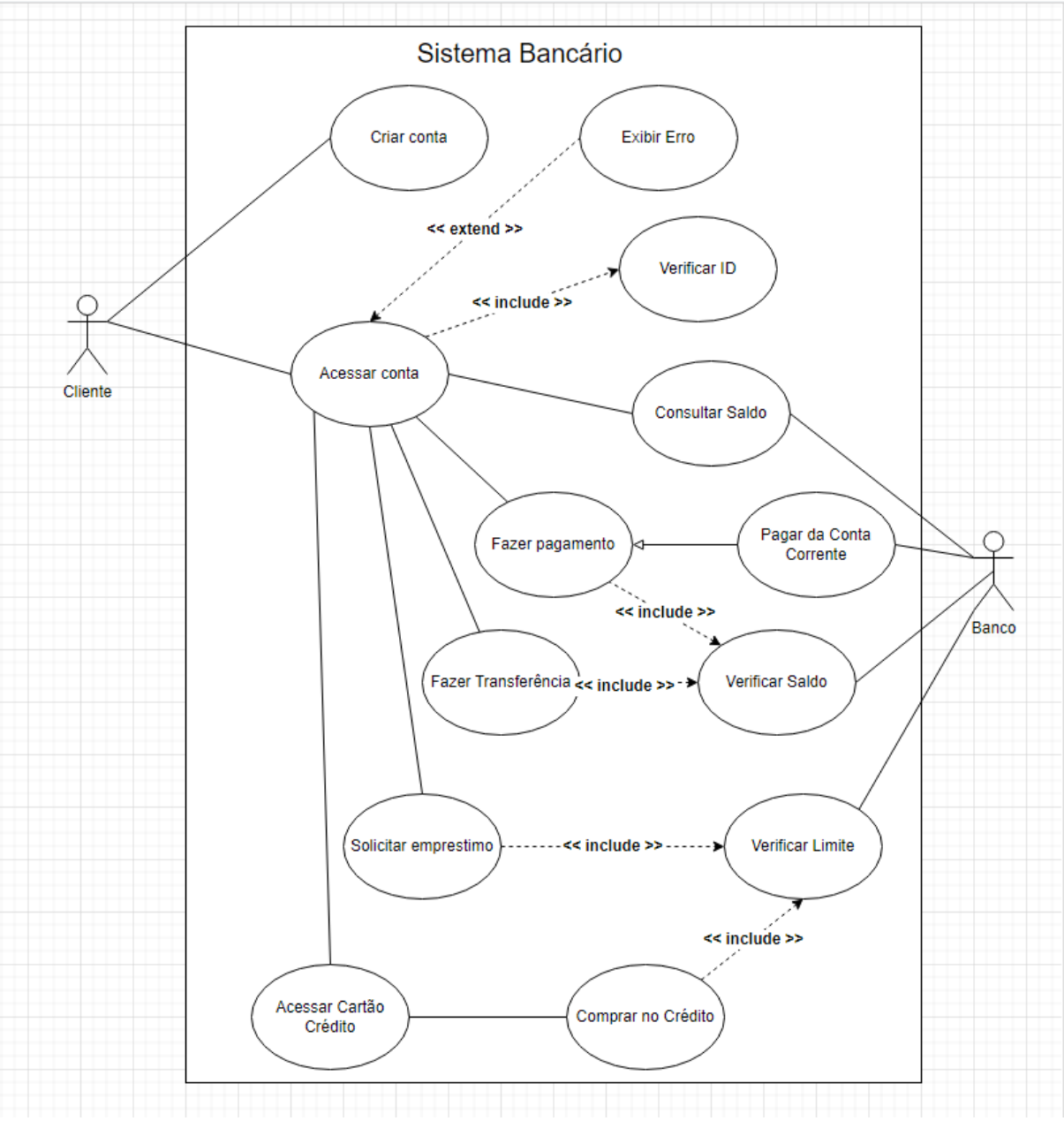
```
public class CartaoCredito extends ContaCorrente{
    double limiteCredito;
    double saldoDisponivel;

    String dataVencimento;
    ArrayList<String> historicoCompras = new ArrayList<>();

    public CartaoCredito(int numConta) {
        super(numConta);
        this.limiteCredito = 0;
        this.saldoDisponivel = 0;
        setFatura(0);
        this.dataVencimento = "0";
    }
```

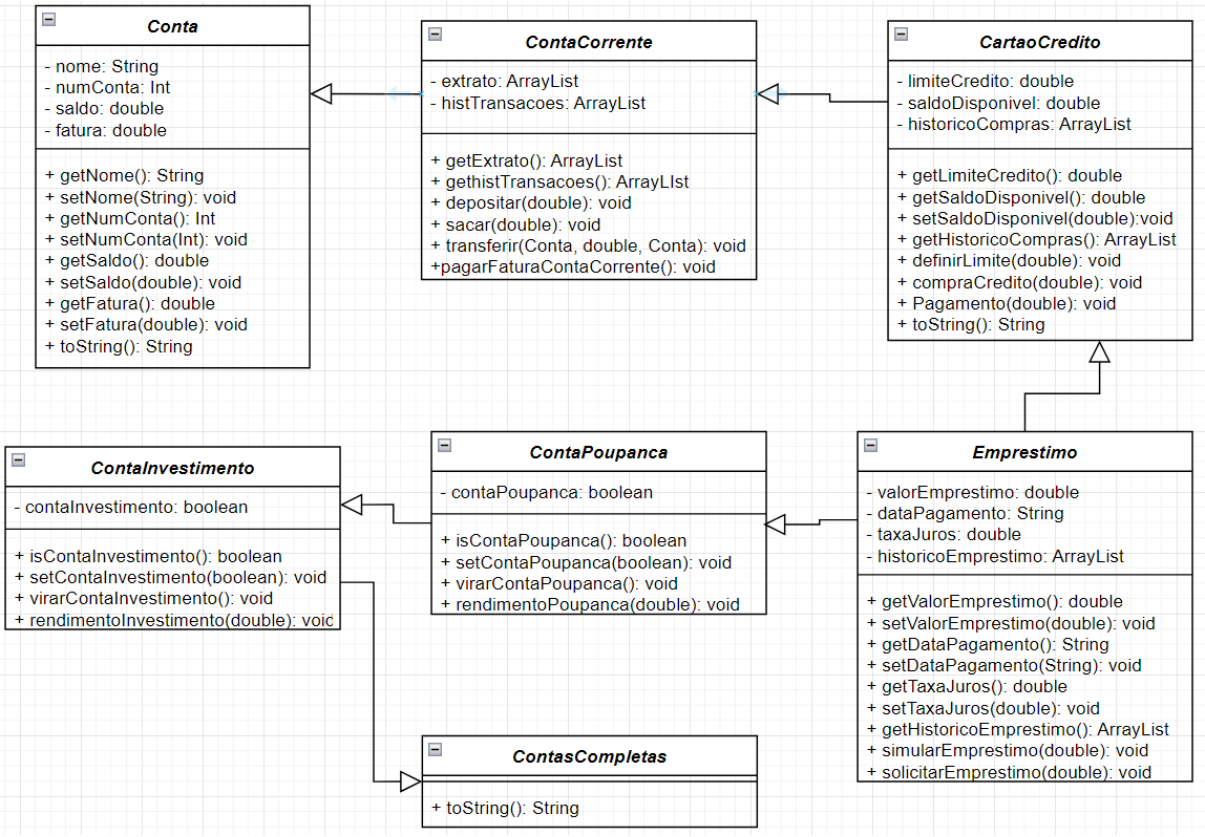
4.1 Diagrama de Caso de Uso (Unidade 1)

Utilizado para uma melhor compreensão dos requisitos e principalmente das interações que o usuário terá com o sistema.



4.2 Diagrama de Classe (Unidade 1)

Utilizado para melhor organização do código e para modelar o sistema que é orientado a objetos. Dá uma melhor visualização para o programador e para os outros membros da equipe se orientarem pelo diagrama.



5 CONSIDERAÇÕES FINAIS (Unidade 2)

O projeto conseguiu completar todos os requisitos que foram solicitados pelo professor, ainda não foi finalizado, pois ainda existem muitas funcionalidades que podem ser implementadas. Porém as funcionalidades que já foram feitas estam todas funcionais, incluindo uma interface que por mais que seja pelo terminal, é de fácil compreensão do usuário final.

5.1 AMEAÇAS AO PROJETO

Quais as dificuldades encontradas que podem invalidar o projeto.

5.2 TRABALHOS FUTUROS

Atualizações no código para fazer mais métodos para a utilização do usuário, como por exemplo um sistema de autenticação, com senha para apenas usuários com a senha conseguir fazer as operações na respectiva conta. Implementar data para fazer com que o usuário pague as contas do cartão de crédito na data específica, e se não ocorrer o pagamento, um juros será cobrado.

Implementar um front-end para a melhor experiência do usuário que seja mais flexível e melhor visívelmente. Usar também o ecossistema Spring para o aumento da produtividade e para conseguir conectar-se melhor a um banco de dados.

REFERÊNCIAS

Neste item será incluída a bibliografia citada (ABNT 6023).
A ordenação dos itens deve ser alfabética. Na mesma referência utiliza-se espaço simples; entre duas referências, um espaço livre.

a) Nas referências:

Todo o documento citado no texto do trabalho deve ser listado abaixo do título REFERÊNCIAS. Como não se trata de elemento textual, não é numerada.
As referências não são justificadas; são alinhadas pela esquerda.
A maneira de referenciar os documentos também deve obedecer às normas da ABNT 6023.
As referências devem ser ordenadas alfabeticamente. Para elaboração do artigo deve se utilizar no mínimo 30 referências.

APÊNDICE/ ANEXOS

Os apêndices/anexos devem vir ao final do trabalho não contando páginas do artigo e devem ser referenciados no corpo do texto (ao final ANEXO A).