SYSPET SONHO: RELATO DE EXPERIÊNCIA SOBRE O DESENVOLVIMENTO DE UM SISTEMA BANCARIO PARA USUARIOS NO PROJETO INTEGRADOR

Analicia Fernandes Aquino Guedes[1]
Daniel Pontes de Queiroz [2]
Vitor Gabriel Mendes Soares [3]
Willames Pereira de Lima [4]

RESUMO

Foi desenvolvido um sistema bancário em Java, com o objetivo de criar uma plataforma segura e intuitiva para clientes gerenciarem suas finanças. Utilizando as IDEs Eclipse e IntelliJ IDEA, foram implementadas classes como cartão de crédito, movimentação, conta corrente e autenticação. O código foi organizado em pacotes, utilizando estruturas condicionais e laços de repetição para interação com o usuário. Foram aplicados tratamentos de exceção para lidar com possíveis erros. O Map foi utilizado para criar uma lista de clientes com chaves primárias. Interfaces interativas foram exibidas com a função JOptionPane.showConfirmDialog(). O sistema proporciona uma experiência segura e eficiente para realizar operações bancárias. Em resumo, o sistema bancário em Java utiliza IDEs populares, conceitos de programação orientada a objetos e ferramentas como Map, resultando em uma plataforma funcional e segura para gerenciamento financeiro.

Palavras-chaves: sistema; java; código.

ABSTRACT

A banking system was developed in Java with the aim of creating a secure and user-friendly platform for clients to manage their finances. Using the Eclipse and IntelliJ IDEA IDEs, classes such as credit card, transaction, checking account, and authentication were implemented. The code was organized into packages, utilizing conditional structures and loops for user interaction. Exception handling was applied to deal with potential errors. The Map data structure was used to create a list of clients with primary keys. Interactive interfaces were displayed using the JOptionPane.showConfirmDialog() function. The system provides a secure and efficient experience for conducting banking operations. In summary, the Java banking system utilizes popular IDEs, object-oriented programming concepts, and tools like Map, resulting in a functional and secure platform for financial management.

1 INTRODUÇÃO

[1] Graduande do Curso de Sistemas para Internet.
[2] Graduande do Curso de Sistemas para Internet.
[3] Graduande do Curso de Sistemas para Internet.
[4] Graduande do Curso de Sistemas para Internet.

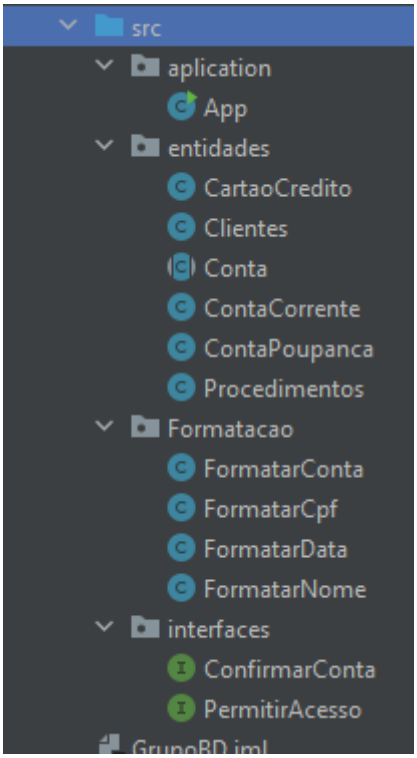
O desenvolvimento de um sistema bancário em Java pode ser um desafio e tanto para qualquer equipe de desenvolvimento. Em particular, a criação de um sistema de usuário que permita aos clientes gerenciar suas contas e realizar transações com segurança requer uma estruturação cuidadosa do código.
Neste relato de experiencia, vamos apresentar o desenvolvimento de um sistema bancário em Java que irá contém classes como cartão de credito, movimentação, conta corrente, autenticação e confirmação de senha, além de uma classe principal. O objetivo deste projeto é criar uma plataforma segura e fácil de usar para clientes bancários, permitindo que eles gerenciem suas finanças com facilidade.
Para isso, será utilizados diversos conceitos importantes da programação orientada a objetos.
Ao longo deste relato de experiência, vamos explorar os desafios encontrados durante o desenvolvimento do sistema, as soluções encontradas e os resultados obtidos.

2 FUNDAMENTAÇÃO TEÓRICA

As ferramentas utilizadas no desenvolvimento do código foram o Eclipse e o IntelliJ IDEA, que são ambientes de desenvolvimento integrado (IDE) populares para a linguagem Java.
O Eclipse é uma IDE de código aberto e amplamente utilizada para desenvolvimento Java. Ele oferece recursos avançados de edição de código, depuração, autocompletar e gerenciamento de projetos. O Eclipse também suporta uma ampla variedade de plugins e extensões que podem ser adicionados para estender suas funcionalidades.
Por outro lado, o IntelliJ IDEA é uma IDE comercial desenvolvida pela JetBrains. Ele também é amplamente usado para desenvolvimento Java e oferece recursos semelhantes ao Eclipse, como edição de código avançada, depuração e gerenciamento de projetos. O IntelliJ IDEA é conhecido por sua interface de usuário intuitiva e amigável, além de fornecer recursos poderosos, como análise estática de código, suporte a testes automatizados e integração com sistemas de controle de versão.
Ambas as IDEs têm uma grande base de usuários e são amplamente adotadas pela comunidade de desenvolvedores Java. Cada uma tem suas próprias características e vantagens, e a escolha entre elas geralmente depende das preferências pessoais e das necessidades do projeto em questão.

3 METODOLOGIA

O código foi dividido em três pacotes para classificar e organizar, que foi o aplication onde ficou a classe App que funciona o código, a seguir o pacote entidades que é onde ficou as classes utilitárias que servem de ferramentas para a classe App e entre as classes ferramentas e a terceira foi a interfaces para guardar as classes ConfirmarConta e PermitirAcesso que servem com interfaces para as classes Clientes, ContaCorrente e ContaPoupança.



Foi utilizado os seguintes comandos para criar uma lista de clientes, e foi utilizado o map para adicionar uma chave primaria para cada cliente.

```
Map<String, List<Clientes>> cliente = new HashMap<>();
```

Também foi usado de artifícios do laço de repetição do while para que o programa funcione até que o usuário clique em sair ou aconteça um erro. Além disto foi usado a função try{código}catch(tipo da exceção){resposta do sistema} e foram implementados tratamentos contra erros de se o usuário usar letras onde é para ser digitados só números entrara em um erro, além do erro que se o usuário não digitar nada entrara em outro erro e se acontecer qualquer outro erro entrará em erro de exceções.

Foi usado estruturas condicionais combinadas em sequencia utilizando o if e o switch, onde o if foi utilizado para casos onde haja condições mais complexas ou poucas saídas, já o switch para casos onde exigem condições simples, mas muitas saídas.

```
while (sairSistema) {

    try{

        int resposta = JOptionPane.showConfirmDialog(null, "Bem vindo! o
senhor já é Cliente? ","Banco Digital", JOptionPane.YES_NO_CANCEL_OPTION);

        if (resposta == 1) {

            int resposta2 = JOptionPane.showConfirmDialog(null, "Deseja
abrir sua conta no banco digital? ","Informação com segurança",
JOptionPane.YES_NO_CANCEL_OPTION);

            if (resposta2 == 0) {//criar conta no aplicativo banco digital

                int opcaoC = JOptionPane.showOptionDialog(null, "Escolha o
tipo de conta:", "Tipo de conta", JOptionPane.DEFAULT_OPTION,
JOptionPane.QUESTION_MESSAGE, null, opcoes, opcoes[0]);
```

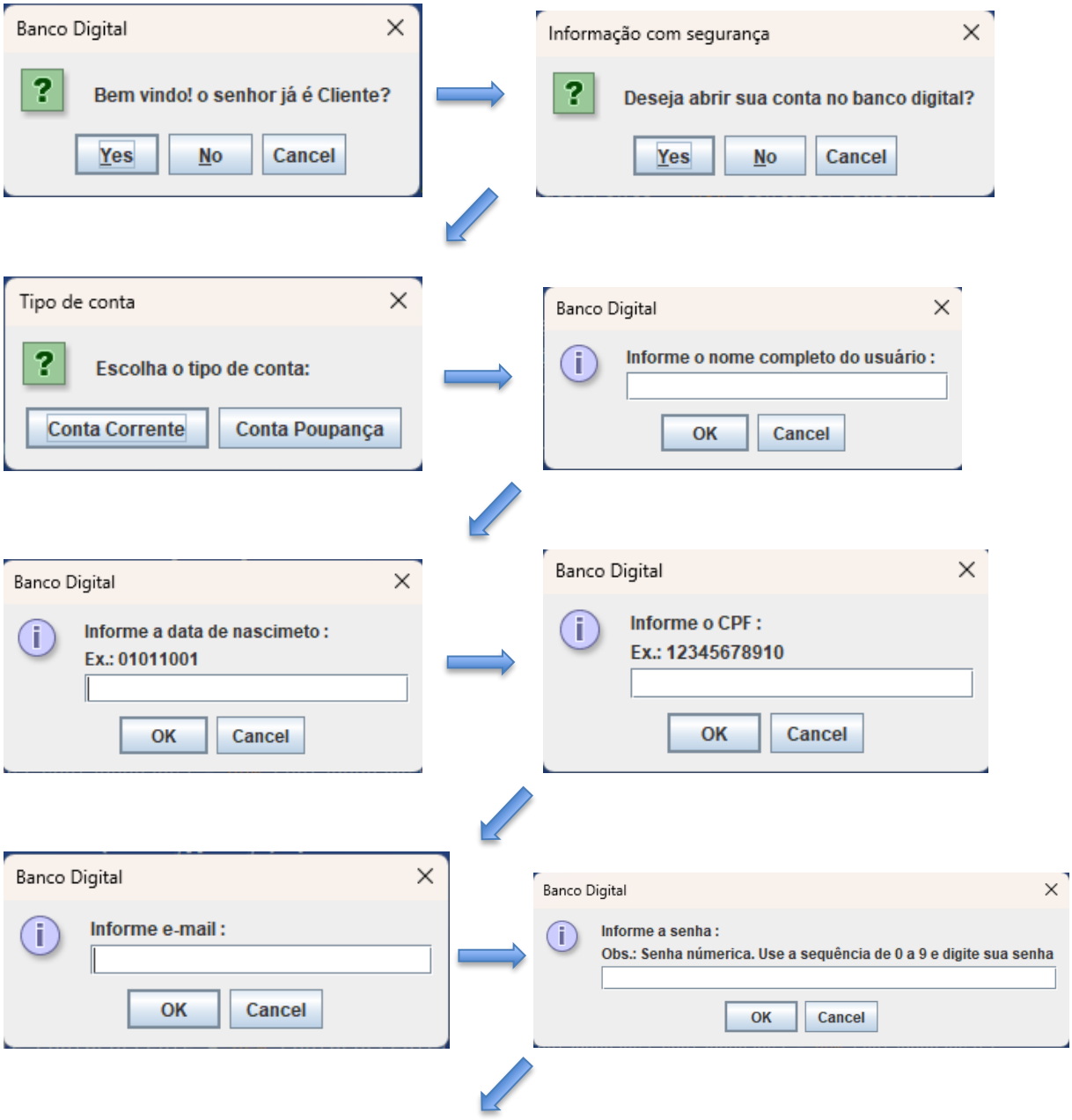
```
int varCc = 0;
int varCp = 0;

switch (opcaoC){
    case 0 -> varCc = 10;
    case 1 -> varCp = 51;
```

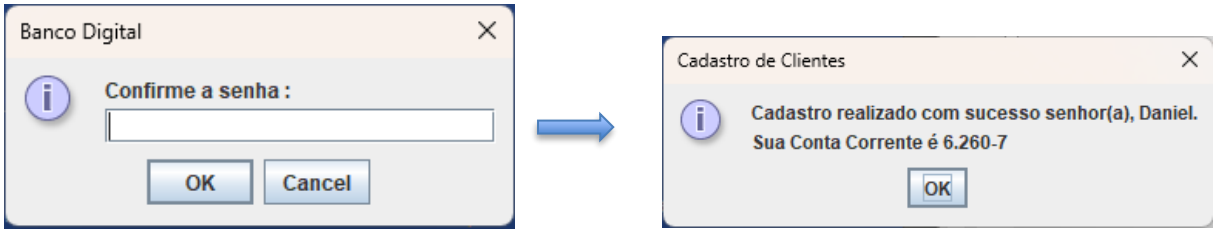
4 RESULTADO E DISCUSSÃO

Para o funcionamento da interface foi utilizado o seguinte comando:

```
JOptionPane.showConfirmDialog()
```

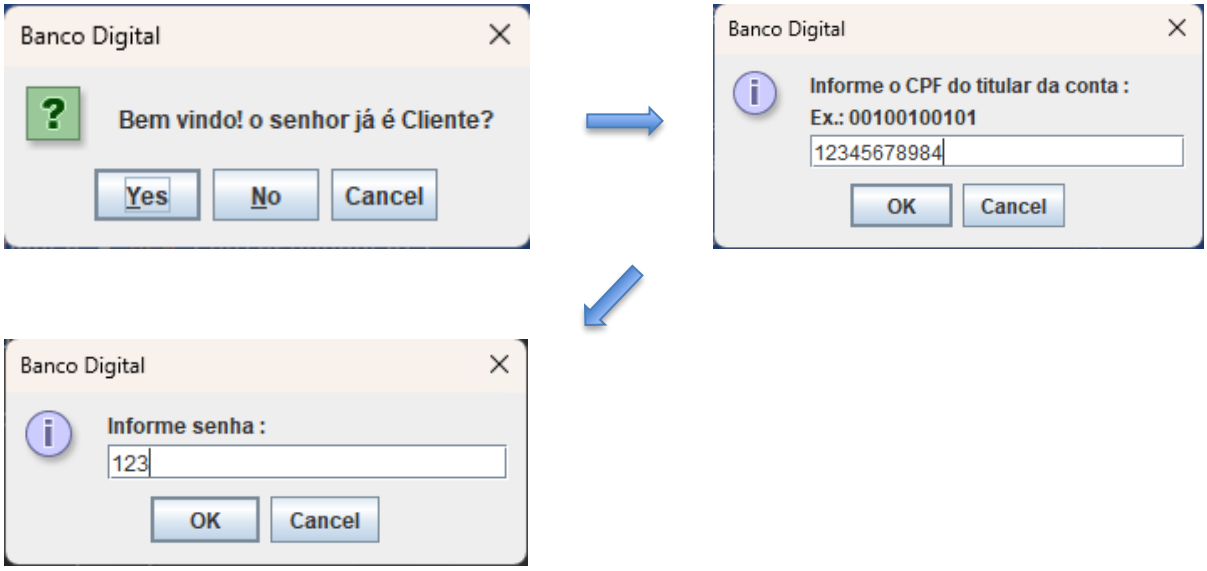
Já para o funcionamento primeiramente foi utilizado uma tela dizendo Bem-vindo! E perguntando se o usuário já é cliente, depois se deseja abrir a conta e em seguida qual o tipo da conta se é conta corrente ou conta poupança. E para o cadastro e pedido ao usuário informações como nome, data de nascimento, cpf, email e a senha que será utilizada para a conta.



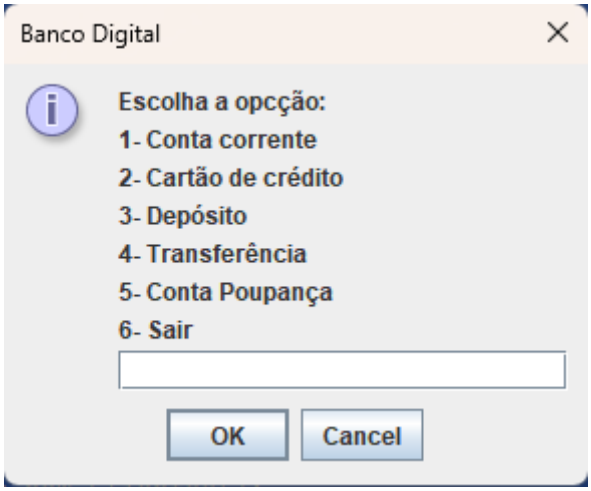


Já para o processo de login foi implementado uma configuração randômica para alternar entre cpf, data de nascimento e número da conta. Onde é pedido ao usuário uma das três opções de forma aleatória e em seguida a senha.

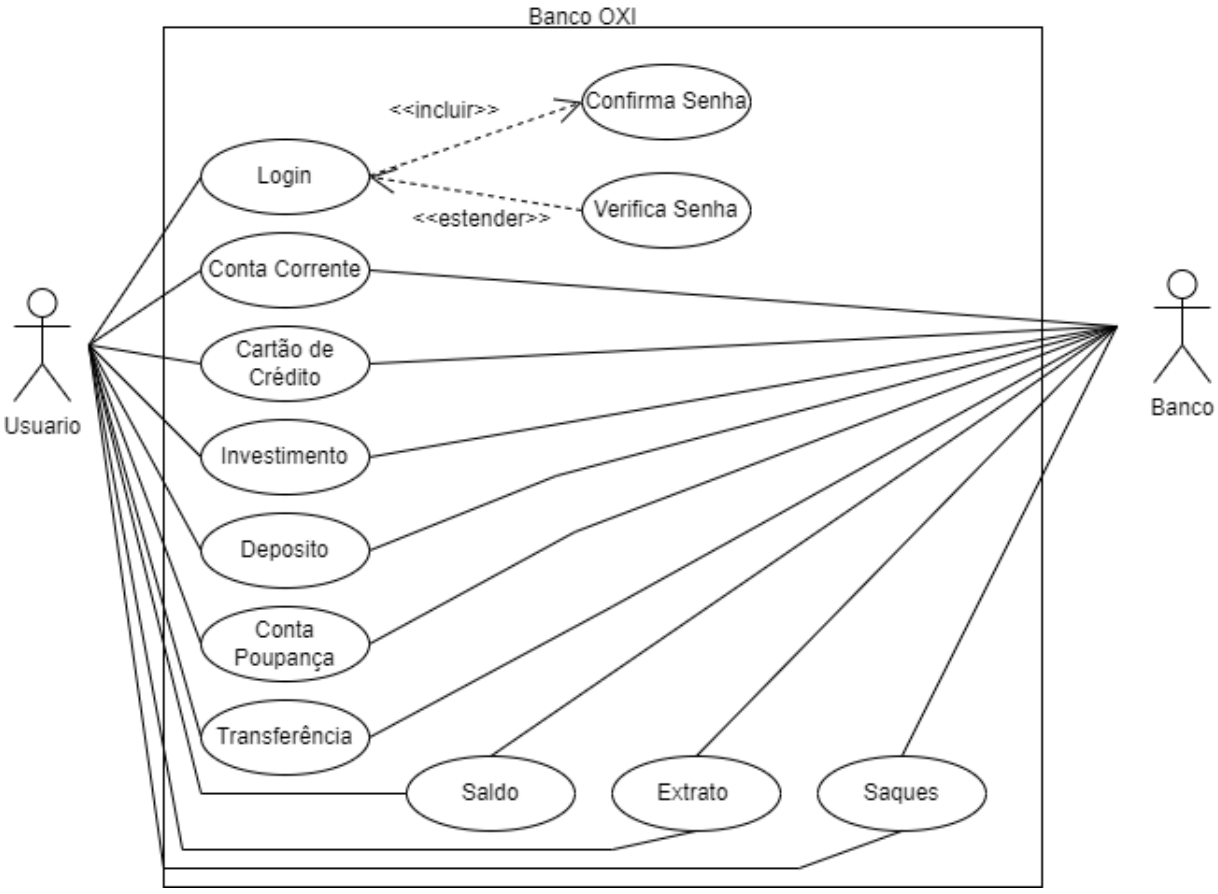
```
int opcaoAcesso = random.nextInt( bound: 3) + 1; // gera um número aleatório entre 1 e 3

FormatarCpf formatarCpf = new FormatarCpf();
FormatarData formatarData = new FormatarData();
FormatarConta formatarConta = new FormatarConta();

//clientes.setPosLogin(opcaoAcesso);//Preciso saber a posicao da escolha para buscar na matriz o dado correto

switch (opcaoAcesso) {
    case 1 -> {
```



Já após o usuário logado a seguintes opções de operações bancarias apareceram, que são Conta corrente, Cartão de credito, Depósito, transferência, Conta poupança e investimento.

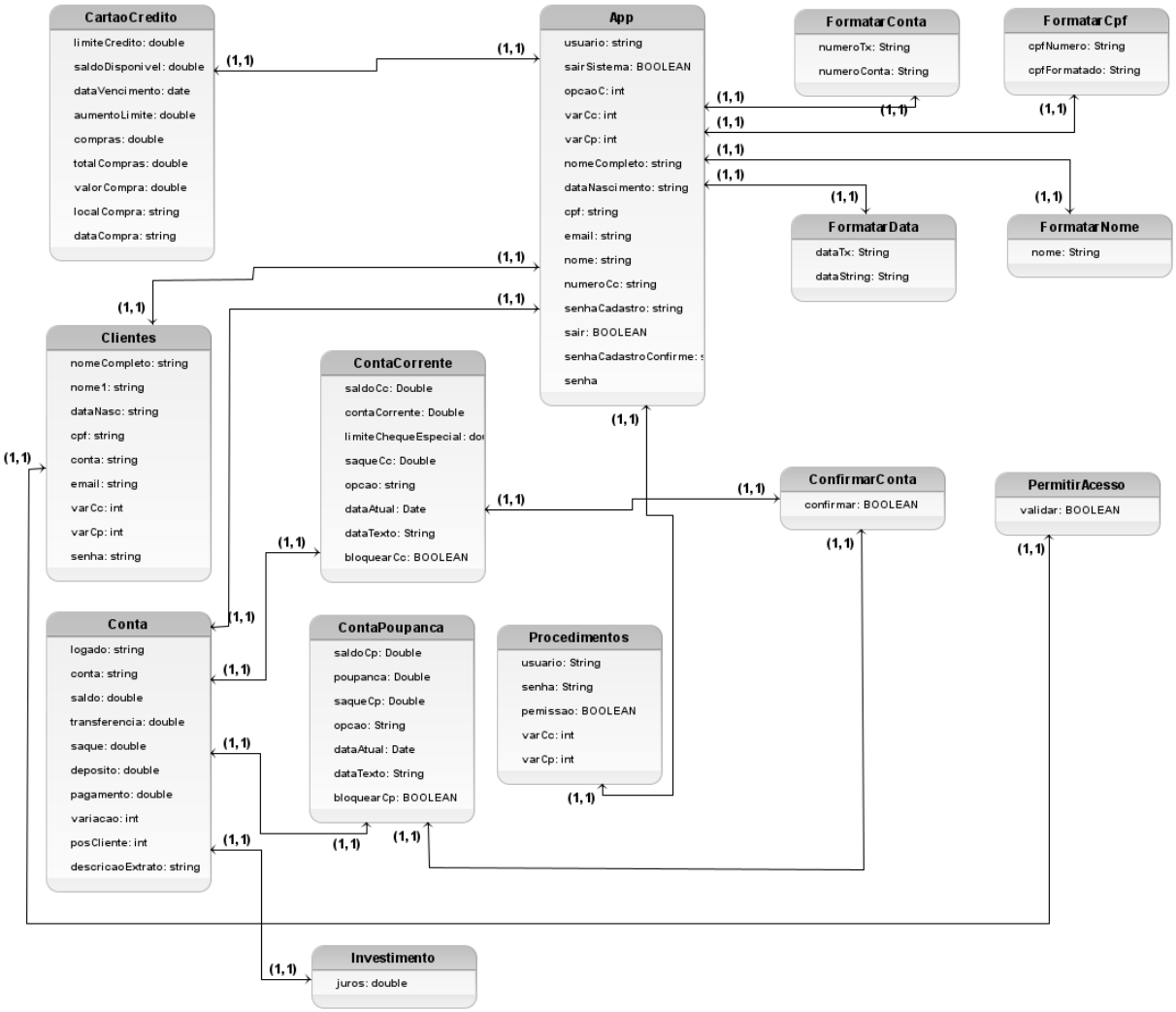


4.1 Diagrama de Caso de Uso



A figura acima ilustra como é a relação entre o usuário e o banco através do sistema.

4.2 Diagrama de Classe



A figura acima consiste em um diagrama de classe, que contém uma classe principal App que acessa as classes CartaoCredito, Cliente, Conta, Procedimentos, FormatarConta, FormatarCpf, FormatarData e FormatarNome. E as classes ContaCorrente e ContaPoupanca herdam da classe Cliente, e também elas têm como interface a classe ConfirmarConta. Já a classe ConfirmaConta é uma interface de cliente.