



# Finanças Pessoais



Simplificando suas economias



# Integrantes

Daniel Augusto

Pedro Medina

Lucas Araújo

Leonardo Magalhães

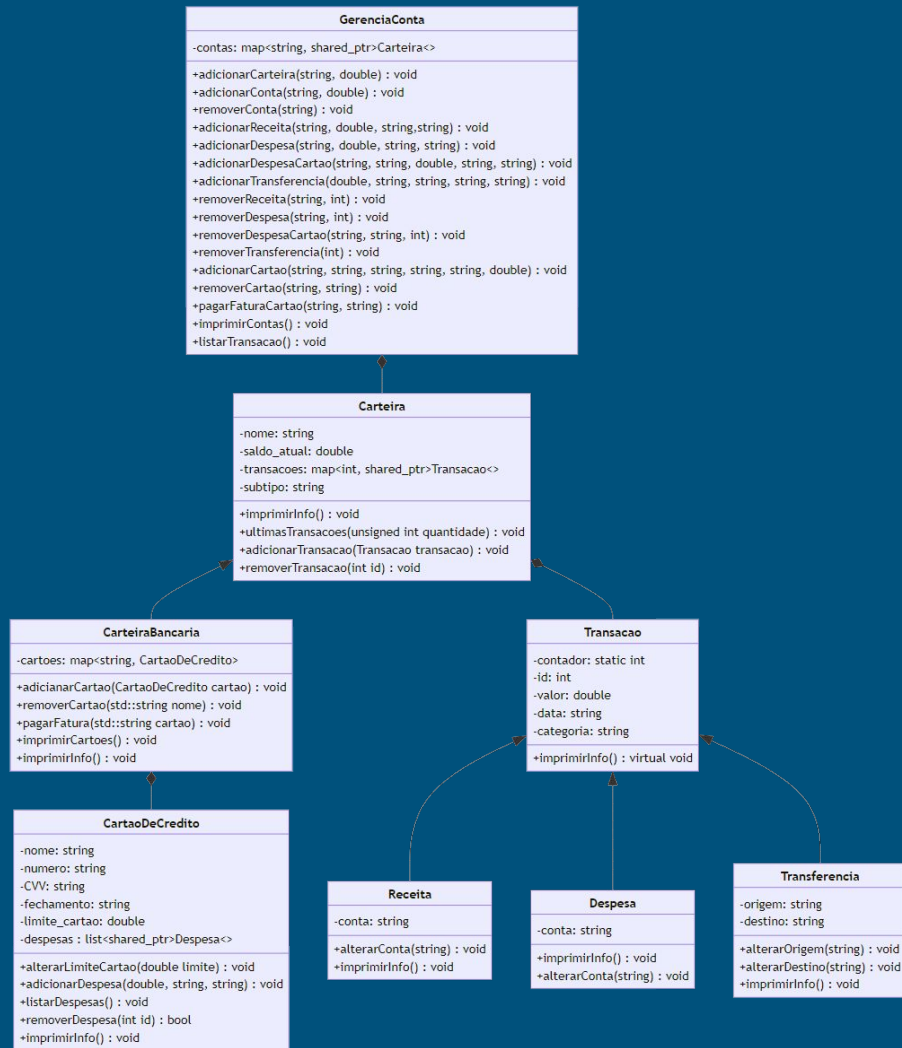
Thiago Silva

---

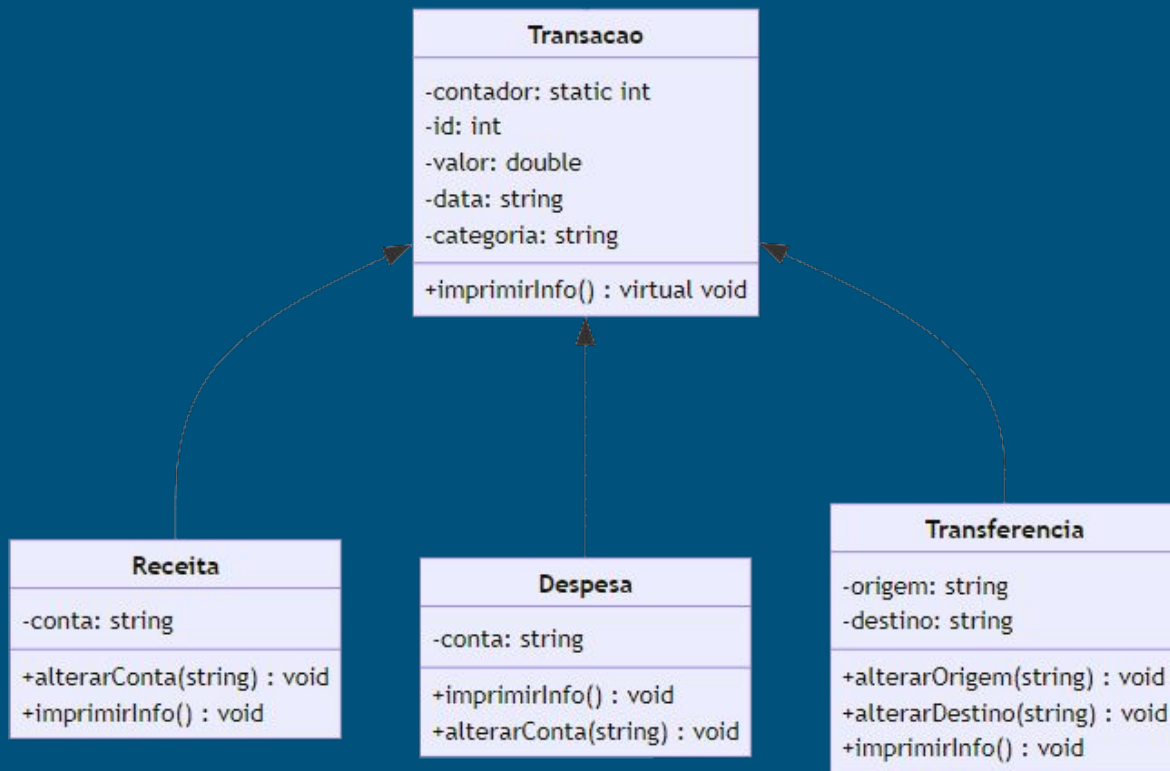
# Objetivos do trabalho

---

- Utilizar os conceitos de herança, encapsulamento e polimorfismo para criar as classes utilizadas para o gerenciamento das finanças.
- Aplicar os princípios do SOLID para ter um código limpo e que seja de fácil extensão.

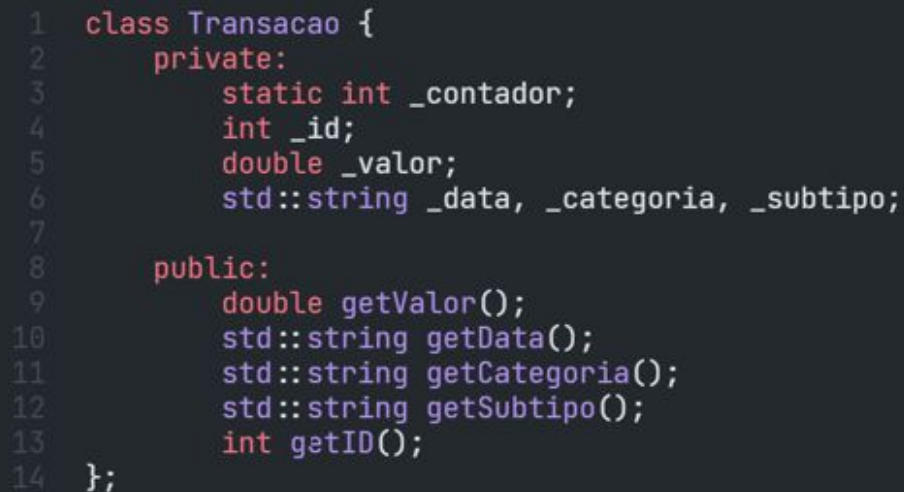


# Herança e Polimorfismo



# Encapsulamento

---



```
1  class Transacao {  
2      private:  
3          static int _contador;  
4          int _id;  
5          double _valor;  
6          std::string _data, _categoria, _subtipo;  
7  
8      public:  
9          double getValor();  
10         std::string getData();  
11         std::string getCategoria();  
12         std::string getSubtipo();  
13         int getID();  
14     };
```

### CarteiraExcp


- ValorInvalido(double valor, std::string nome)
- ContaNaoEncontrada(std::string nome)
- ContaJaExiste(std::string nome)
- ContaNaoPermiteCartao(std::string nome\_conta, std::string tipo\_conta)
- SaldoInsuficiente(double saldo, double despesa)

### CartaoDeCreditoExcp

- +LimiteExcedido(std::string nome\_cartao, std::string numero\_cartao, double limite\_cartao, double soma\_despesas\_atuais)
- +LimiteInvalido(std::string nome\_cartao, std::string numero\_cartao, double limite\_cartao)
- +CartaoNaoEncontrado(std::string nome\_cartao)
- +NumeroInvalido(std::string numero)
- +CVVInvalido(std::string CVV)
- +FechamentoInvalido(std::string fechamento)
- +CartaoJaExiste(std::string nome)

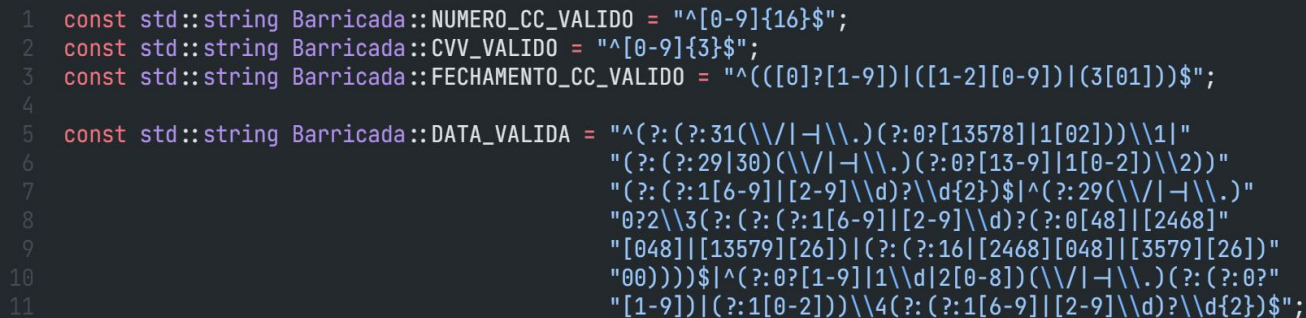
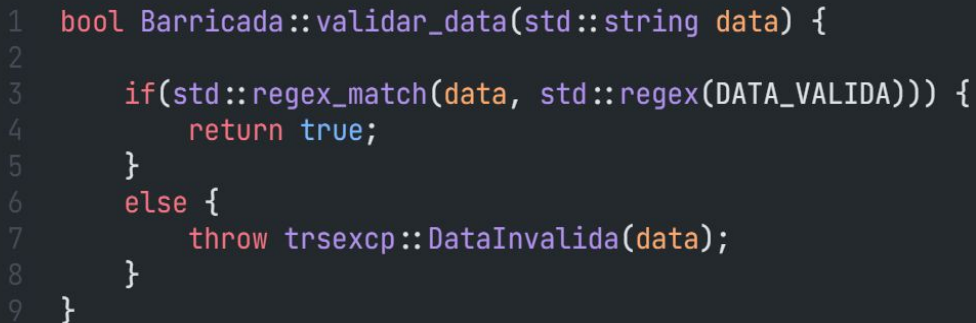
### TransacaoExcp

- TransacaoNaoEncontrada(int id)
- TipoTransacaoInvalido(std::string tipo)
- DataInvalida(std::string data)
- TransferencialInvalida(std::string)



```
1  class Barricada {
2      private:
3          static const std::string NUMERO_CC_VALIDO;
4          static const std::string CVV_VALIDO;
5          static const std::string FECHAMENTO_CC_VALIDO;
6          static const std::string DATA_VALIDA;
7
8      public:
9          static void validar_saldo(double &saldo);
10
11         static void validar_input(unsigned &input);
12
13         static void validar_transacao(double &valor_transacao);
14
15         static void validar_id(unsigned &id);
16
17         static void validar_limite_cartao(double &limite);
18
19         static bool validar_cartao(std::string numero, std::string CVV, std::string fechamento);
20
21         static bool validar_transferencia(std::string data, std::string origem, std::string destino);
22
23         static bool validar_data(std::string data);
24     };
```





# Principal desafio

Robustez do programa



# Perguntas?