

RELATÓRIO SISTEMA BANCÁRIO

Aluna: Maria Eduarda Gobbi

Turma: 1F

OBJETIVO DA ADAPTAÇÃO

O objetivo principal foi reorganizar o código para utilizar classes, encapsulamento, modularização e boas práticas de OO, mantendo as funcionalidades originais do sistema (cadastro de contas, login, depósitos, saques, transferências, auditoria e área do gerente).

2. ESTRUTURA DO PROJETO

A estrutura final do projeto foi organizada da seguinte forma: sistema_bancario/ • account.py – Define a classe Account, responsável por representar cada conta bancária. • transaction.py – Contém a classe Transaction, que registra cada operação realizada. • bank.py – Implementa a classe Bank, que gerencia contas, transações e regras do sistema. • cli.py Parte responsável pela interface no terminal (menus e interação com o usuário). run.py – Arquivo principal, utilizado para iniciar o sistema.

3. PRINCIPAIS MUDANÇAS REALIZADAS

3.1 Classe Account: A classe Account representa individualmente cada conta bancária.

Foi implementado:

- Encapsulamento da senha do usuário;
- Métodos para autenticação, depósito, saque, alteração de senha e controle de ativação;
- Tratamento financeiro com arredondamento adequado.

3.2 Classe Transaction: A classe Transaction registra todos os detalhes das operações:

- Timestamp.
- Tipo de operação.
- Conta de origem e destino.
- Valor, taxa e saldos finais.

3.3 Classe Bank: A classe Bank centraliza a lógica do sistema:

- Armazena e gerencia contas.
- Aplica taxas de forma automatizada.
- Realiza depósitos, saques e transferências com validações.
- Implementa o controle do gerente, com listagem de saldos.

- Mantém histórico de todas as transações.

3.4 Interface via CLI ([cli.py](#)): criado para separar regras de negócio da parte visual. Ele contém:

- `menu_principal()` – para criar contas, login e acesso ao gerente.
- `menu_conta()` – para operações internas do usuário autenticado.

3.5 Arquivo `run.py`: cria uma instância de `Bank` e chama o menu principal. Execução do sistema pelo terminal.

CONCLUSÃO

A adaptação para Programação Orientada a Objetos tornou o sistema mais sólido, bem estruturado. Todas as funcionalidades originais foram preservadas, e o código agora segue boas práticas de modularização, reaproveitamento e encapsulamento.