

**Brasília - DF**

**2024**

**Pró-Reitoria Acadêmica**

**Escola de Educação, Tecnologia e Comunicação**

**Curso de Bacharelado em Engenharia de Software**

**Trabalho da Disciplina de Laboratório de Banco de Dados**

**BANCO MALVADER**

**Autores: Guilherme Fiorelli Braz Rodrigues, João Gabriel Brito Checchia.**

**Orientador: Prof. William Roberto Malvezzi**

**Guilherme Fiorelli Braz Rodrigues**

**João Gabriel Brito Checchia**

**Banco Malvader**

Documento apresentado ao Curso de graduação de Bacharelado em Engenharia de Software da Universidade Católica de Brasília, como requisito parcial para obtenção da aprovação na disciplina de Laboratório de Banco de Dados.

Orientador: Prof. William Roberto Malvezzi.

**Brasília**

**2024**

**RESUMO**

Referência: Fiorelli, Guilherme Braz Rodrigues; BRITO, João Gabriel Checchia. Banco Malvader, 2024. 46p. Bacharelado em Engenharia de Software – UCB – Universidade Católica de Brasília, Taguatinga – DF, 2024.

O objetivo deste trabalho é desenvolver um sistema bancário simples, com a implementação de funcionalidades essenciais, utilizando a linguagem de programação Java. O sistema tem como foco a simulação de operações bancárias como abertura de conta, depósito, saque e consulta de saldo, além de permitir a criação de múltiplas contas e o gerenciamento de seus respectivos saldos. A aplicação foi construída utilizando uma arquitetura orientada a objetos, visando a eficiência e a clareza no código. O sistema é de fácil utilização, com uma interface simples que facilita a interação do usuário, e foi desenvolvido para ser utilizado em ambientes acadêmicos, sendo uma ferramenta didática que auxilia na compreensão de conceitos fundamentais de programação em Java, como classes, objetos e manipulação de dados.

Palavras-chave: Sistema Bancário, Java, Orientação a Objetos, Depósito, Saque, Programação.

**ABSTRACT**

This work aims to develop a simple banking system with the implementation of essential features, using the Java programming language. The system focuses on simulating banking operations such as account creation, deposit, withdrawal, and balance inquiry, as well as allowing the creation of multiple accounts and the management of their respective balances. The application was built using an object-oriented architecture, aiming for efficiency and clarity in the code. The system is easy to use, with a simple interface that facilitates user interaction, and was developed for use in academic environments, serving as an educational tool to aid in the understanding of fundamental programming concepts in Java, such as classes, objects, and data manipulation.

Keywords: Banking System, Java, Object-Oriented, Deposit, Withdrawal, Programming.

**SUMÁRIO**

**1. INTRODUÇÃO**

**2. OBJETIVO**

2.1 Objetivo Geral

2.2 Objetivos Específicos

**3**. **ABRANGÊNCIA DO SISTEMA**

3.1 Regras de Negócio

3.1 Requisitos funcionais

3.2 Requisitos não funcionais

3.3 Metodologia e Tecnologia Utilizada

**4. DOCUMENTAÇÃO E ANÁLISE**

4.3 Diagrama de atividades do sistema

4.1 Diagrama de Caso de Uso Geral

4.2 Diagramas de Caso de Uso Específicos

4.3 Especificação dos Casos de Uso

4.4 Diagramas de Sequências

4.5 Diagrama de Classe

4.6 Modelo lógico

4.7 Dicionário de Dados

**5. PROJETO DE INTERFACES**

5.1 Tela Inicial

5.2 Tela Login

5.3 Telas da funcionalidade principal

**6. CONCLUSÃO**

**7. GLOSSÁRIO**

**8. REFERÊNCIAS BIBLIOGRÁFICAS**

1. **INTRODUÇÃO**

O sistema de gerenciamento bancário "Banco Malvader" foi desenvolvido com o intuito de proporcionar um ambiente seguro e eficiente para a administração de contas bancárias e de clientes. Utilizando a linguagem de programação Java e a interface gráfica Swing, o projeto busca integrar funcionalidades essenciais para a realização de operações bancárias, como abertura de contas, consulta de saldo, depósitos e saques, entre outras. Além disso, o sistema também permite a geração de relatórios financeiros e o controle administrativo por parte dos funcionários.

Este sistema foi projetado com base em conceitos de Programação Orientada a Objetos (POO) e implementa persistência de dados através de um banco de dados MySQL, permitindo que as informações sejam armazenadas e recuperadas conforme necessário. O sistema é estruturado para oferecer uma experiência de usuário intuitiva e segura, com autenticação e funcionalidades específicas para clientes e funcionários.

1. **OBJETIVO**

## 2.1 Objetivo Geral

Desenvolver um sistema bancário completo utilizando Java, com interface gráfica Swing e persistência de dados em um banco de dados MySQL. O objetivo principal é fornecer uma plataforma que permita o gerenciamento de contas bancárias, o cadastro de clientes e funcionários, além da realização de operações financeiras como depósitos, saques e consultas de saldo.

2.2 Objetivos Específicos

* Implementar um sistema de login com autenticação de usuários, com opções para funcionários e clientes.
* Permitir a abertura e o encerramento de contas bancárias, tanto para contas poupança quanto correntes.
* Oferecer funcionalidades para consulta de dados, como saldo, limite disponível, extrato bancário e dados de clientes.
* Implementar a possibilidade de gerar relatórios financeiros e exportá-los para formatos como Excel.
* Prover uma interface gráfica com Swing, garantindo uma interação amigável com o usuário.
* Implementar uma camada de persistência de dados utilizando um banco de dados MySQL, com a criação das tabelas necessárias e o gerenciamento de transações.

1. **ABRANGÊNCIA DO SISTEMA**

3.1 Regras de Negócio

* Cadastro e gerenciamento de contas: O sistema deve permitir que os funcionários do banco abram novas contas, tanto poupança quanto corrente. Além disso, o sistema deve possibilitar o fechamento de contas por meio de uma autenticação administrativa.
* Operações financeiras: Clientes podem consultar seu saldo, realizar depósitos e saques, verificar limites de crédito e consultar extratos. Os saques devem ser verificados com base no saldo disponível, e depósitos devem ser registrados corretamente.
* Relatórios financeiros: O sistema permitirá que os funcionários gerem relatórios de movimentações bancárias, que serão exportáveis para Excel.
* Consultas e alterações: O sistema deve permitir a consulta e alteração de dados tanto de clientes quanto de funcionários, com a validação de senhas para realizar modificações sensíveis.

3.2 Requisitos funcionais

* Autenticação de usuários: Implementação de login com senha, diferenciando entre funcionários e clientes.
* Abertura e encerramento de contas: O funcionário pode abrir e encerrar contas bancárias, com possibilidade de editar dados de contas existentes.
* Operações bancárias: O cliente pode realizar operações como consulta de saldo, depósitos, saques e obter extratos.
* Cadastro de funcionários e clientes: Funcionários podem cadastrar novos usuários, tanto clientes quanto outros funcionários.
* Geração de relatórios financeiros: Os funcionários podem gerar relatórios detalhados sobre movimentações bancárias e exportá-los.
* Alteração de dados: Funcionalidades para alteração de dados de clientes e funcionários, com verificação de permissões.

3.3 Requisitos não funcionais

* Segurança: O sistema deve garantir a segurança das transações, com autenticação de usuários e criptografia de senhas.
* Desempenho: O sistema deve ser rápido e responsivo, com tempos de resposta reduzidos para operações bancárias e consultas.
* Usabilidade: A interface gráfica deve ser intuitiva e de fácil navegação para garantir uma boa experiência para o usuário final.
* Persistência de dados: Todos os dados do sistema devem ser persistidos em um banco de dados MySQL, garantindo integridade e recuperação em futuras execuções do sistema.

3.4 Metodologia e Tecnologia Utilizada

* Metodologia: O desenvolvimento do sistema será realizado utilizando uma abordagem ágil, com iterações semanais para garantir que os requisitos sejam atendidos ao longo do desenvolvimento. Testes unitários serão feitos para validar a integridade do sistema.
* Tecnologias:
  + Java: A linguagem principal para o desenvolvimento do sistema.
  + Swing: Para a criação da interface gráfica.
  + MySQL: Para o gerenciamento de dados e persistência do sistema.
  + JDBC: Para a integração entre o Java e o banco de dados MySQL.
  + Excel (APACHE POI): Para exportação de relatórios financeiros.

1. **DOCUMENTAÇÃO E ANÁLISE**

4.1 Diagrama de atividades do sistema

4.2 Diagrama de Caso de Uso Geral

O Diagrama de Caso de Uso Geral descreve as interações principais entre os **atores** (usuários ou sistemas externos) e o sistema "Banco Malvader". Aqui estão os principais casos de uso que o sistema oferece:

* **Atores:**
* **Funcionário**: Um funcionário do banco, responsável por gerenciar contas, realizar cadastro de funcionários, gerar relatórios, etc.
* **Cliente**: O usuário que possui uma conta bancária e pode acessar funcionalidades como consultar saldo, realizar depósitos e saques, entre outras.
* **Administrador**: (Implícito, geralmente o papel de quem tem acesso total ao sistema, como o gerente de banco).
* **Casos de Uso:**

1. **Acesso ao Sistema**
   * Autenticação inicial com opções para funcionário ou cliente.
   * Entrada com senha.
2. **Menu Funcionário**
   * Abertura de Conta.
   * Encerramento de Conta.
   * Consulta de Dados (de clientes e contas).
   * Alteração de Dados.
   * Cadastro de Funcionários.
   * Geração de Relatórios.
3. **Menu Cliente**
   * Consultar Saldo.
   * Realizar Depósito.
   * Realizar Saque.
   * Consultar Extrato.
   * Consultar Limite.
4. **Operações Administrativas**
   * Gerenciamento de contas bancárias.
   * Geração de relatórios.
5. **Encerrar Sistema**:
   * Finalização da aplicação.

Diagrama

Descrição gerada automaticamente

4.3 Diagramas de Caso de Uso Específicos

4.4 Especificação dos Casos de Uso

1. Abertura de Conta (Funcionário)

* Objetivo: Permitir ao funcionário abrir uma nova conta bancária para um cliente.
* Ator Principal: Funcionário.
* Pré-condição: O funcionário está autenticado no sistema.
* Fluxo Principal:
  + Funcionário escolhe a opção "Abertura de Conta".
  + O sistema solicita o tipo de conta (Poupança ou Corrente).
  + O funcionário escolhe o tipo de conta.
  + O sistema solicita os dados do cliente (nome, CPF, endereço, telefone etc.).
  + O funcionário insere as informações.
  + O sistema valida os dados.
  + O sistema cria a conta e associa o cliente a ela.
  + O sistema exibe uma mensagem de sucesso.
* Fluxos Alternativos:
  + Se os dados fornecidos são inválidos (ex: CPF já cadastrado), o sistema informa o erro e permite nova tentativa.
* Pós-condição: Uma nova conta foi criada e associada ao cliente.

2. Consulta de Saldo (Cliente)

* + Objetivo: Permitir que o cliente consulte o saldo de sua conta bancária.
  + Ator Principal: Cliente.
  + Pré-condição: O cliente está autenticado no sistema.
  + Fluxo Principal:
    - Cliente escolhe "Consultar Saldo" no menu.
    - O sistema solicita a senha do cliente.
    - O cliente insere sua senha.
    - O sistema válida a senha.
    - O sistema exibe o saldo da conta do cliente.
  + Fluxos Alternativos:
    - Se a senha fornecida for inválida, o sistema exibe uma mensagem de erro e permite nova tentativa.
  + Pós-condição: O saldo da conta do cliente é exibido.

4.5 Diagramas de Sequências

4.6 Diagrama de Classe

Diagrama

Descrição gerada automaticamente

4.7 Modelo lógico

Diagrama

Descrição gerada automaticamente com confiança média

4.8 Dicionário de Dados

1. **PROJETO DE INTERFACES**

5.1 Tela Inicial

5.2 Tela Login

5.3 Telas da funcionalidade principal

1. **CONCLUSÃO**
2. **GLOSSÁRIO**
3. **REFERÊNCIAS BIBLIOGRÁFICAS**