Universidade Federal de Pernambuco - Centro de Informática
Curso de Sistemas de Informação
Disciplina: Estrutura de Dados Orientadas a Objetos

Sistema Bibliotecário - CRUD

Integrantes: Giovanna Clócate, Mateus Ribeiro e Sérgio Lira.
03/02/2025

Introdução

Sabemos que Sistemas informatizados tornaram-se indispensáveis para facilitar o controle de acervos, gerenciar usuários e automatizar processos, garantindo eficiência e acessibilidade.

Este projeto apresenta um sistema de gerenciamento bibliotecário em CRUD. Desenvolvido em C++ utilizando os princípios da orientação a objetos (OO) para gerenciar livros e usuários, permitindo a realização de empréstimos e devoluções. O objetivo principal do sistema é proporcionar uma plataforma organizada e eficiente para o gerenciamento de bibliotecas.

O sistema foi projetado para atender bibliotecas com o escopo de pequeno a médio porte, permitindo idealmente, um administrador realizar o cadastro, busca, atualização e deleção (CRUD) de livros e usuários. Além disso, conta com um mecanismo de autenticação, garantindo que apenas usuários autorizados realizem ações específicas, como empréstimos de livros e administração do acervo.

Conforme orientado durante as aulas da disciplina de EDOO, a estrutura do sistema é baseada em classes e herança. A classe base **Livro** é utilizada para representar as características comuns a todos os livros, sendo posteriormente especializadas nas classes **LivroFisico** e **LivroDigita**l. Da mesma forma, **Usuario** é a classe base para os perfis de **Leitor Comum** e **Administrador**, garantindo um controle adequado dos acessos e permissões dentro do sistema.

O desenvolvimento seguiu uma **abordagem incremental**, garantindo que cada funcionalidade fosse implementada e testada antes da adição de novos recursos. Segue na sequência apresentação da arquitetura do sistema, apresentando os principais conceitos de herança, polimorfismo e encapsulamento, além de diagramas UML que ilustram a estrutura e as relações entre as classes**.**

Arquitetura do Sistema

Como mencionado anteriormente, a arquitetura do sistema segue um modelo baseado em entidades centrais, como usuários, livros e empréstimos, estas interagem entre si por meio de uma classe controladora, o **Sistema**.

- Visão Geral da Arquitetura

O sistema é composto por quatro divisões principais:

1. Controle (**Sistema**): Responsável por gerenciar a lógica do sistema, incluindo cadastro, busca, empréstimos e autenticação de usuários.
2. Modelos (**Livro, Usuario, Emprestimo**): Representa os principais objetos do sistema, utilizando herança e polimorfismo para especializar classes.
3. Persistência (arquivos de texto): Armazena os dados de usuários e livros de forma persistente.
4. Interface (**Menu**): Permite a interação do usuário com o sistema, processando entradas e exibindo informações. (Vale notar que o Front-End será mais detalhado em sua seção própria)

O link interativo para diagrama UML abaixo ilustra a arquitetura geral do sistema e as relações entre as classes:

https://lucid.app/lucidchart/3cb6b549-92ac-4d61-bae2-d5c7146bb15b/edit?viewport_loc=-959%2C-2139%2C7453%2C3581%2CHWEp-vi-RSFO&invitationId=inv_907deba6-9235-4222-bedc-db7782af95a0

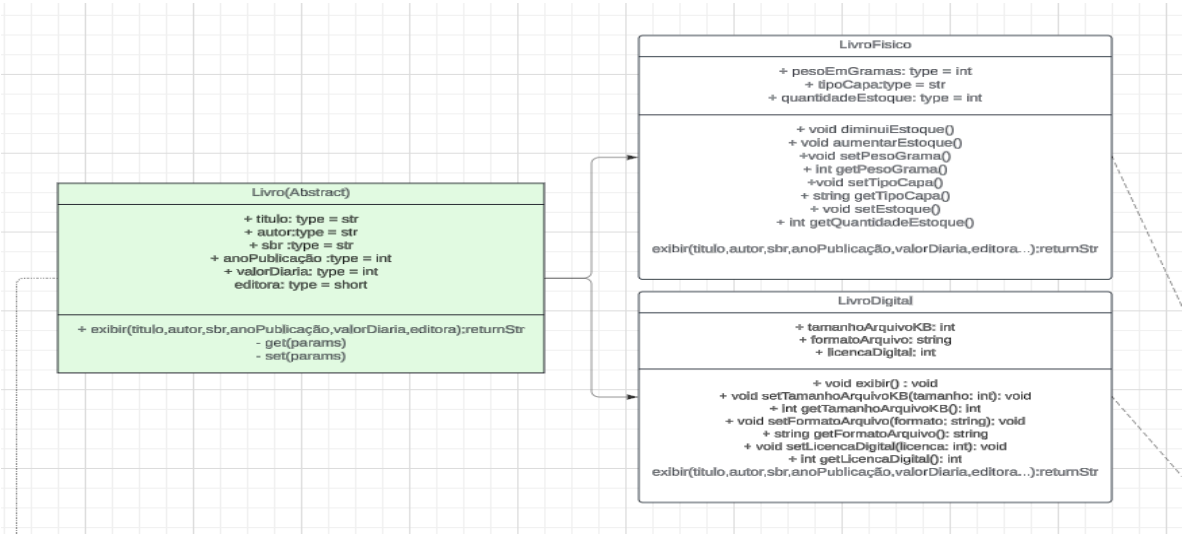
- Estrutura de Classes e Herança

Hierarquia da Classe Livro

A classe Livro representa qualquer livro cadastrado na biblioteca. Ela contém atributos essenciais, como título, autor, ISBN, ano de publicação e editora. A partir dela, foram especializadas duas classes:

- **LivroFisico**: Representa livros físicos, adicionando atributos como peso e tipo de capa.
- **LivroDigita**l: Representa livros digitais, incluindo informações como tamanho do arquivo e formato.

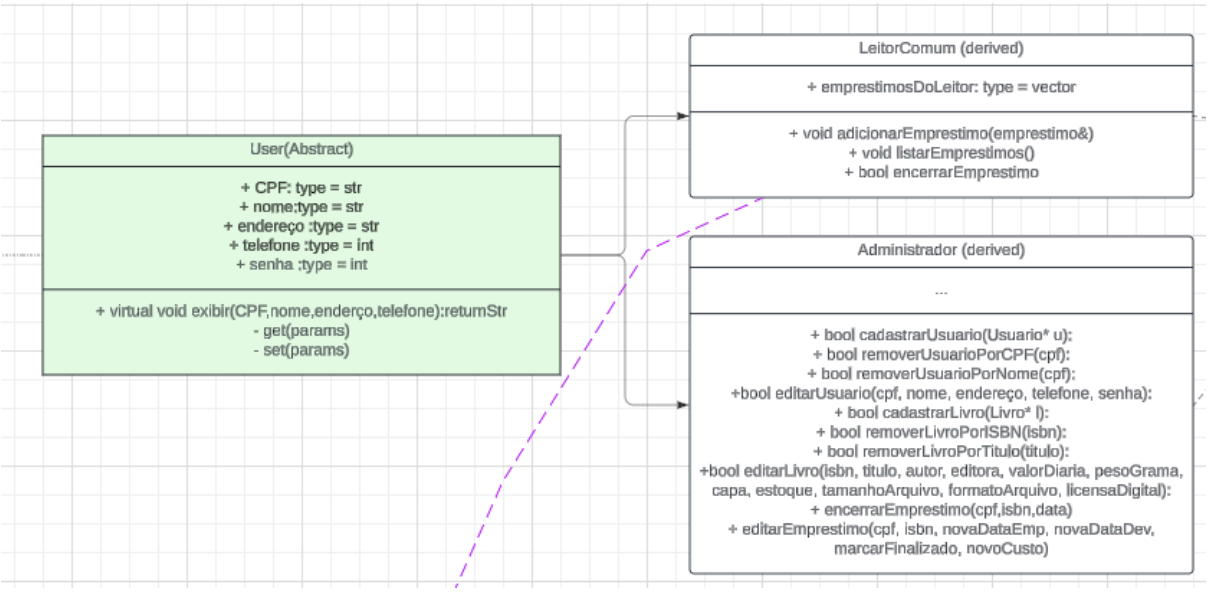
Diagrama UML da Classe Livro e herdeiras:



Hierarquia da Classe <u>Usuario</u>

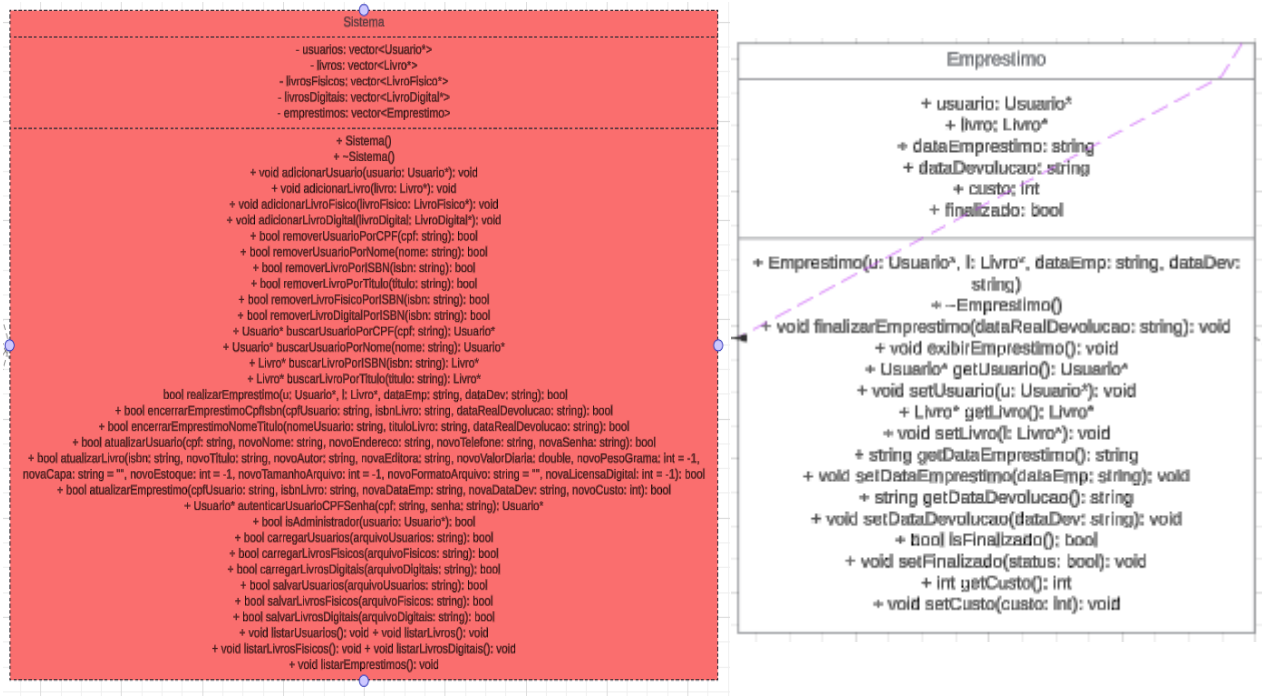
A classe Usuario define as informações básicas de um usuário, como nome, CPF e endereço. Ela é utilizada como classe base para dois perfis diferentes:

- **LeitorComum**: Pode buscar livros, realizar empréstimos e devolver livros.
- **Administrador**: Tem permissões avançadas, podendo cadastrar, remover e atualizar livros e usuários.



Classe Controladora (Sistema) e Classe Relacional (Empréstimo)

A classe Sistema é responsável por gerenciar a lógica do sistema e conectar os diferentes componentes. Ela mantém listas de usuários, livros e empréstimos, além de fornecer métodos para CRUD e controle de empréstimos. Já a classe Emprestimo, é responsável por criar um dado que interliga um livro a um usuário e determina imediatamente o custo do empréstimo, podendo o mesmo ser encerrado quando necessário.

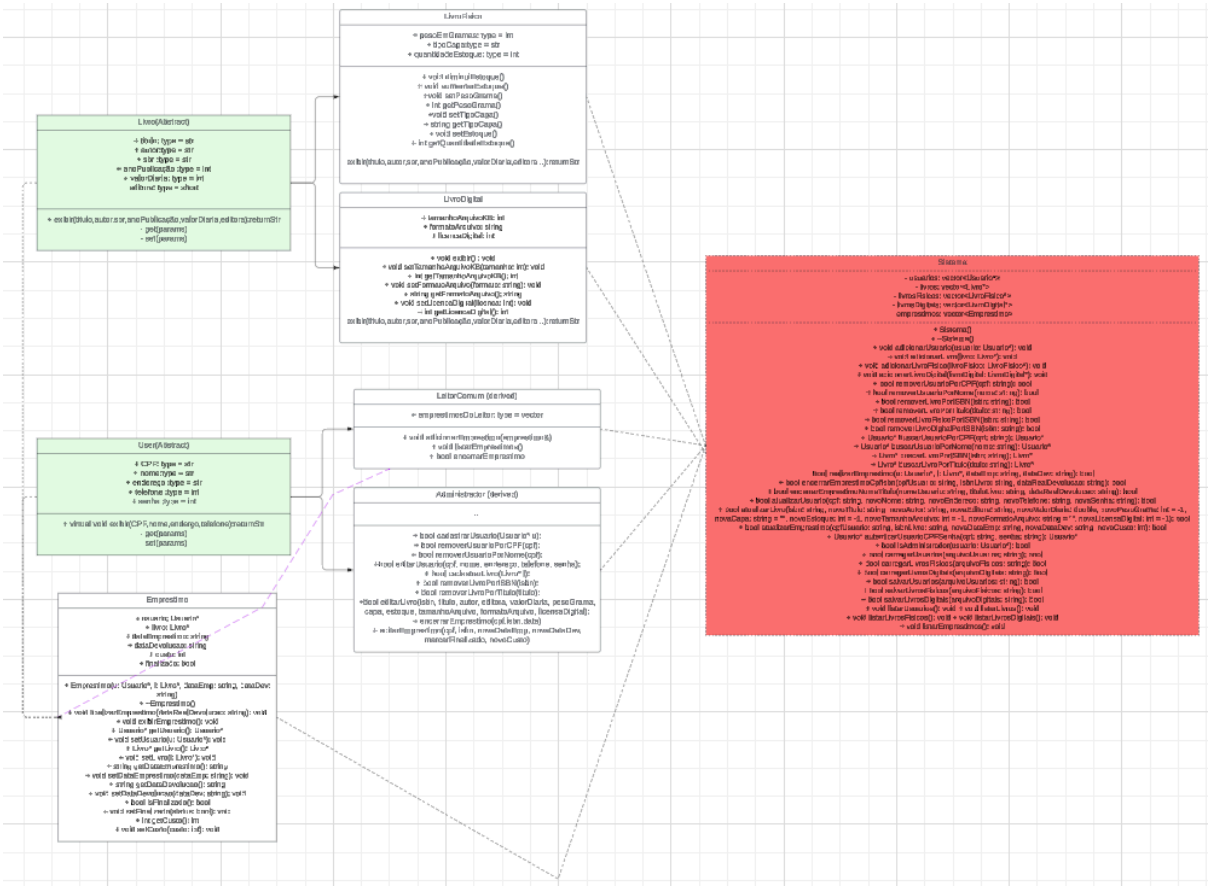


- Fluxo de Execução do Sistema

O funcionamento do sistema segue um fluxo estruturado:

1. O usuário faz login (CPF e senha são verificados).
2. Caso seja um administrador, ele pode cadastrar, remover ou editar usuários e livros.
3. Caso seja um leitor comum, ele pode buscar livros, alugar e devolver.
4. Cada empréstimo é registrado, vinculando um usuário a um livro por um período definido.
5. O sistema mantém listas organizadas e permite buscas eficientes.

Segue em seguida o diagrama completo:



Padrões de Orientação a Objetos Utilizados

O sistema utiliza três pilares da orientação a objetos:

- Encapsulamento: As informações são protegidas por meio de modificadores de acesso (private e protected).
- Herança: Reutiliza código através das classes base Usuario e Livro.
- Polimorfismo: Métodos como exibir() são redefinidos para cada tipo de livro (LivroFisico e LivroDigital).

- Funcionalidades implementadas:
- **CRUD (Create, Read, Update, Delete)**
- **Autenticação**
    - Verificação de Login
- **Sistema de Empréstimos**
    - Ligação entre classes para criar um emprestimo

Interface Gráfica

Para proporcionar uma experiência mais amigável ao usuário, o sistema foi desenvolvido com **Qt Creator**, um framework moderno e eficiente para criação de interfaces gráficas em C++.

A interface gráfica do sistema foi projetada para ser **intuitiva e completa**, oferecendo os seguintes elementos:

Menu Principal – Permite que usuários comuns realizem o login ou cadastro e o administrador acessem o sistema com CPF e senha.

Tela de Cadastro – Permite que os usuários comuns realizem o cadastro ao informar o nome, endereço, telefone e senha.
Tela de Login – Permite que usuários ou o administrador acessem o sistema com CPF e senha.

Menu do Usuário Comum – Fornece as opções de visualizar os livros cadastrados na biblioteca e os empréstimos realizados pelo usuário.

- Tela de Visualizar Livros – Mostra os livros físicos e digitais da biblioteca.
- Tela de Visualizar Empréstimos – Mostra todos os empréstimos ativos, concluídos e vencidos do usuário.

Menu do Administrador – Mostra opções de gerenciamento de livros físicos e digitais, usuários e empréstimos.

Paineis de Gerenciamento – Administradores podem **cadastrar, editar, pesquisar e remover livros e empréstimos** através de botões e formulários. Já no painel de usuários o administrador não pode adicionar, já que a tabela é atualizada após receber um novo cadastro no sistema.

O Qt Creator facilita a separação entre **lógica de interface e lógica de negócio**, utilizando o **padrão MVC (Model-View-Controller)**:

- **Model:** Representa os dados do sistema (livros, usuários, empréstimos).
- **View:** Define os componentes visuais (botões, tabelas, formulários).
- **Controller:** Processa as interações do usuário e executa as ações adequadas no sistema.

OBS: Embora o sistema tenha uma interface gráfica mais moderna, ele também pode ser executado via **terminal**.

Contribuições

Giovanna Clócate: Atuou especialmente na organização de arquivos e desenvolvimento do Front-End bem como na criação do banco de dados no SQLite e na integração do código bruto com a interface gráfica.

Mateus Ribeiro: Atuou especialmente organizando e desenvolvendo o Back-End, organização inicial de arquivos, criação e integração de classes e funções.

Sérgio Lira: Atuou auxiliando tanto o desenvolvimento do Front-End como do Back-End, testes, correções e desenvolvimento de funções.

Considerações Finais

O desenvolvimento proporcionou uma experiência de aprendizado na aplicação dos conceitos de **orientação a objetos (OO)** em C++. Durante a construção do projeto, foi possível adquirir conhecimentos sobre herança, polimorfismo, encapsulamento e definitivamente sobre boas práticas de modularização, bem como um melhor entendimento da linguagem de C++.

A implementação da interface gráfica com **Qt Creator** foi um dos maiores desafios, pois exigiu um entendimento aprofundado do framework e da integração entre os componentes visuais e a lógica do sistema, que se apresentou como um tanto difícil sem experiência prévia. Além disso, a manipulação de **listas dinâmicas, e principalmente gerenciamento de memória e ponteiros** apresentou dificuldades iniciais, mas que foram superadas.

Ao final do projeto, foi possível compreender melhor como estruturar um sistema completo, definitivamente lições foram tiradas sobre o **planejamento da arquitetura** até a **implementação de funcionalidades e interface**. Acreditamos que a experiência vai ajudar inclusive a novos projetos futuros.