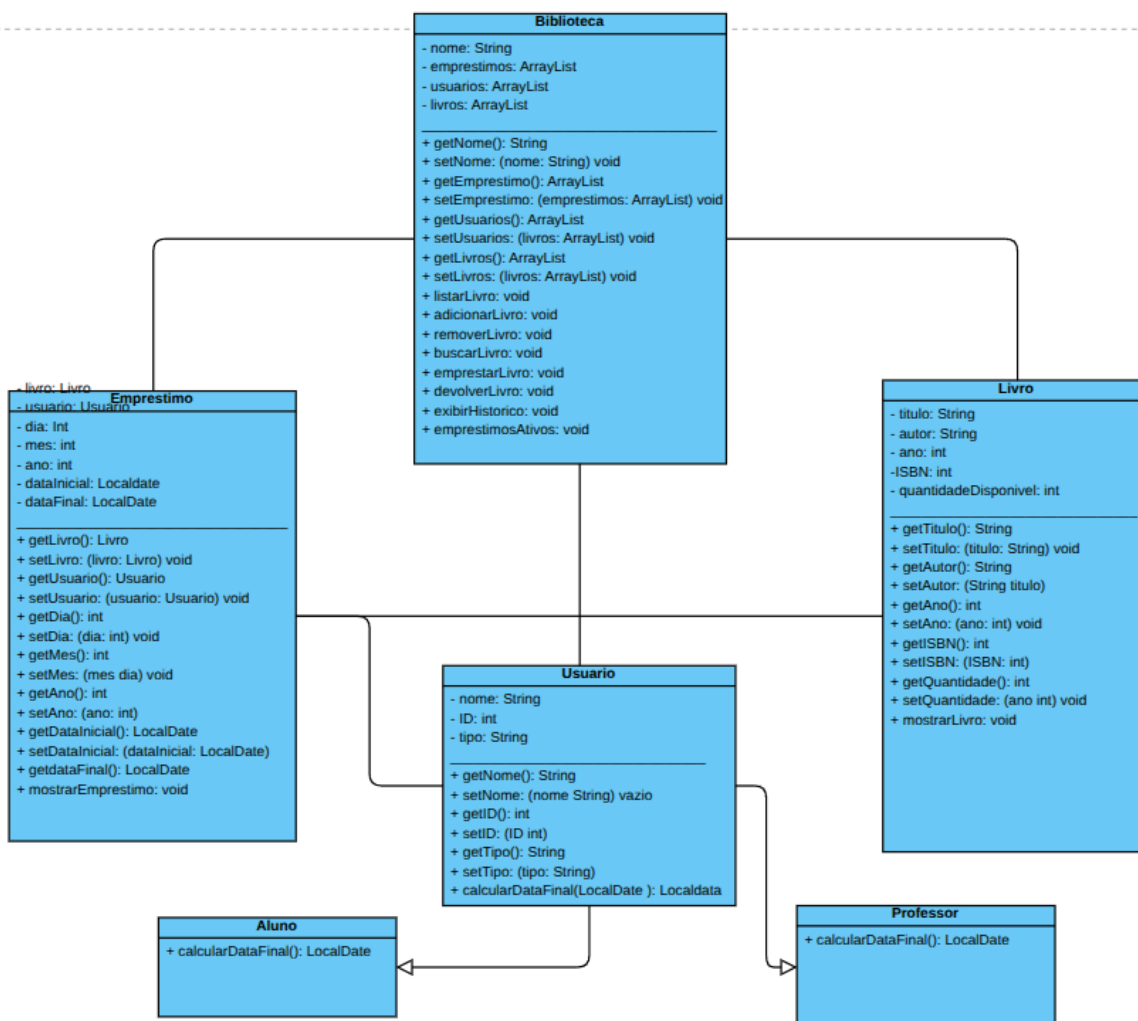


1. DESCRIÇÃO DO SISTEMA E SEUS OBJETIVOS:

O sistema desenvolvido neste TDE se trata de um sistema para o controle e gerenciamento de uma biblioteca. Seus objetivos são: registrar as entradas e saídas de livros na biblioteca com os métodos: adicionarLivro() e removerLivro, criação de objetos: Livro, Usuario (classe pai) que se deriva em duas sub-classes Professor e Aluno, Empréstimo e Biblioteca, exibir livros existentes na biblioteca com o método: listarLivros(), procurar se determinado livro existe na biblioteca em questão com o método: buscarLivro(), emprestar livros por um período determinado com o método: emprestarLivro(), devolver os livros emprestados com o método: devolverLivro(), exibir histórico de empréstimos de um usuário com o método: exibirHistorico() e visualizar empréstimos de livros que estão ativos com o método: emprestimosAtivos();

2. DIAGRAMA DE CLASSES



3. PRINCIPAIS DECISÕES

Esse projeto envolveu diversas camadas de desenvolvimento e validações, onde tarefas aparentemente simples se tornaram complexas ao exigir verificações que refletissem situações reais. A implementação de funcionalidades como controle de empréstimos, distinção entre perfis de usuários e cálculo de prazos demandou grande atenção às regras de negócio, resultando em um sistema mais fiel à dinâmica de uma biblioteca de verdade. As principais decisões foram principalmente como fazer as verificações e quais classes seriam atributos de outras, onde a primeiro momento achei que poderia fazer de determinada maneira e percebi que não daria certo, precisando assim retomar tudo de volta.

4. EVIDÊNCIAS DE EXECUÇÃO

4.1 FOTOS DO CÓDIGO MAIN

```

1 package app;
2 import service.Biblioteca;import model.Livro;import model.Professor; import model.Aluno;
3
4 public class Main {
5     public static void main(String[] args) throws InterruptedException {
6
7         // Criando service.Biblioteca
8         Biblioteca BibliotecaPuc = new Biblioteca( nomeBiblioteca: "service.Biblioteca PUC");
9
10        // Criando Livros
11        Livro l1 = new Livro( titulo: "Percy Jackson e o Ladrão de Raios", autor: "Rick Riordan", ano: 2005, isbn: 1, quantidadeDisponivel: 1);
12        Livro l2 = new Livro( titulo: "Percy Jackson e o Mar de Monstros", autor: "Rick Riordan", ano: 2006, isbn: 2, quantidadeDisponivel: 4);
13        Livro l3 = new Livro( titulo: "Percy Jackson e a Maldição do Titã", autor: "Rick Riordan", ano: 2007, isbn: 3, quantidadeDisponivel: 6);
14        Livro l4 = new Livro( titulo: "Percy Jackson e a Batalha do Labirinto", autor: "Rick Riordan", ano: 2008, isbn: 3, quantidadeDisponivel: 6);
15
16        // Criando Usuarios
17        Aluno u1 = new Aluno( nome: "Rafael", id: 1, tipo: "Aluno");
18        Aluno u2 = new Aluno( nome: "Abner", id: 2, tipo: "Aluno");
19        Professor u3 = new Professor( nome: "Matheus", id: 3, tipo: "Professor");
20
21        // Adicionando livros a biblioteca
22        System.out.println("Adicionando Livros: ");
23        BibliotecaPuc.adicionarLivro(l1);
24        BibliotecaPuc.adicionarLivro(l2);
25        BibliotecaPuc.adicionarLivro(l3);
26        BibliotecaPuc.adicionarLivro(l4);
27
28        // Listando livros da biblioteca
29        System.out.println("Lista de Livros: ");
30        BibliotecaPuc.listarLivros();
31
32        // Empréstando livros
33        BibliotecaPuc.emprestarLivro(l1, u2, dia: 4, mes: 5, ano: 2007);

```

```

4 public class Main {
5     public static void main(String[] args) throws InterruptedException {
6
7         // Adicionando livros a biblioteca
8         System.out.println("Adicionando Livros: ");
9         BibliotecaPuc.adicionarLivro(l1);
10        BibliotecaPuc.adicionarLivro(l2);
11        BibliotecaPuc.adicionarLivro(l3);
12        BibliotecaPuc.adicionarLivro(l4);
13
14        // Listando livros da biblioteca
15        System.out.println("Lista de Livros: ");
16        BibliotecaPuc.listarLivros();
17
18        // Empréstando livros
19        BibliotecaPuc.emprestarLivro(l1, u2, dia: 4, mes: 5, ano: 2007);
20
21        // Exemplo de livro que vai dar errado o emprestimo
22        BibliotecaPuc.emprestarLivro(l1, u2, dia: 4, mes: 8, ano: 2007);
23        BibliotecaPuc.emprestarLivro(l3, u3, dia: 10, mes: 9, ano: 2025);
24        BibliotecaPuc.emprestarLivro(l2, u3, dia: 4, mes: 8, ano: 2024);
25
26        // Exibindo histórico de empréstimos dos usuarios
27        System.out.println("Exibindo Históricos: ");
28        BibliotecaPuc.exibirHistorico(u3);
29        BibliotecaPuc.exibirHistorico(u1);
30
31        // Devolvendo livro
32        BibliotecaPuc.devolverLivro(l2);
33
34        // Buscando Livro
35        BibliotecaPuc.buscarLivro( tituloLivro: "Percy Jackson e a Maldição do Titã");
36
37        // Verificando empréstimos ativos

```

```
4 public class Main {
5     public static void main(String[] args) throws InterruptedException {
6
7         // Emprestando livros
8         BibliotecaPuc.emprestarLivro(l1, u2, dia: 4, mes: 5, ano: 2007);
9
10        // Exemplo de livro que vai dar errado o emprestimo
11        BibliotecaPuc.emprestarLivro(l1, u2, dia: 4, mes: 8, ano: 2007);
12        BibliotecaPuc.emprestarLivro(l3, u3, dia: 10, mes: 9, ano: 2025);
13        BibliotecaPuc.emprestarLivro(l2, u3, dia: 4, mes: 8, ano: 2024);
14
15        // Exibindo histórico de empréstimos dos usuarios
16        System.out.println("Exibindo Históricos: ");
17        BibliotecaPuc.exibirHistorico(u3);
18        BibliotecaPuc.exibirHistorico(u1);
19
20        // Devolvendo livro
21        BibliotecaPuc.developerLivro(l2);
22
23        // Buscando Livro
24        BibliotecaPuc.buscarLivro(tituloLivro: "Percy Jackson e a Maldição do Titã");
25
26        // Verificando empréstimos ativos
27        BibliotecaPuc.emprestimosAtivos();
28
29        // removendo livro
30        BibliotecaPuc.removerLivro(l4);
31    }
32}
```

4.2 FOTOS DA EXECUÇÃO NO TERMINAL

```
Run Main x
/usr/java/jdk-24-oracle-x64/bin/java -javaagent:/snap/intellij-idea-community/649/lib/idea_rt.jar=37757 -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8
Adicionando Livros:
Livro Percy Jackson e o Ladrão de Raios adicionado com sucesso!
Livro Percy Jackson e o Mar de Monstros adicionado com sucesso!
Livro Percy Jackson e a Maldição do Titã adicionado com sucesso!
Não é possível adicionar esse livro pois já existe um outro com esse mesmo ISBN
Lista de livros:
Nossos títulos presentes nessa biblioteca são:
1. Percy Jackson e o Ladrão de Raios
2. Percy Jackson e o Mar de Monstros
3. Percy Jackson e a Maldição do Titã
Usuário tentando emprestar um livro...
Livro emprestado com sucesso!
Usuário tentando emprestar um livro...
Todos os livros deste título já foram emprestados, aguarde ficar disponível ou escolha outro título
Usuário tentando emprestar um livro...
Livro emprestado com sucesso!
Usuário tentando emprestar um livro...
Livro emprestado com sucesso!
Exibindo Históricos:
0 livro emprestado foi: Percy Jackson e a Maldição do Titã
```

```
Main.java x Biblioteca.java Aluno.java Emprestimo.java Livro.java Usuario.java Professor.java
Run Main x
usuario tentando emprestar um livro...
Livro emprestado com sucesso!

Exibindo Históricos:
O livro emprestado foi: Percy Jackson e a Maldição do Titã
O usuário que emprestou o livro foi: Matheus
Esse empréstimo iniciou dia: 10/09/2025
Esse empréstimo finalizará dia: 10/10/2025

O livro emprestado foi: Percy Jackson e o Mar de Monstros
O usuário que emprestou o livro foi: Matheus
Esse empréstimo iniciou dia: 04/08/2024
O empréstimo deste livro finalizou dia: 03/09/2024

Esse usuário não efetuou nenhum empréstimo.

Devolvendo livro...
O livro só pode ser devolvido após o pagamento da multa por extrapolar a data

O título Percy Jackson e a Maldição do Titã existe em nossa biblioteca.

O livro emprestado foi: Percy Jackson e a Maldição do Titã
O usuário que emprestou o livro foi: Matheus
Esse empréstimo iniciou dia: 10/09/2025
Esse empréstimo finalizará dia: 10/10/2025
Estado do empréstimo: ativo

Process finished with exit code 0
```

4.3 FOTOS DOS CÓDIGOS DAS CLASSES

```
Sistema-Gerenciador-de-Biblioteca-main Version control Main
Main.java Biblioteca.java Aluno.java Emprestimo.java Livro.java Usuario.java Professor.java

1 package service;
2 import java.time.LocalDate;import java.util.ArrayList;import model.*;
3
4 public class Biblioteca { 3 usages
5     private String nomeBiblioteca; 4 usages
6     private ArrayList<Emprestimo> empréstimos = new ArrayList<>(); 6 usages
7     private ArrayList<Livro> livros = new ArrayList<>(); 10 usages
8     private ArrayList<Usuario> usuarios = new ArrayList<>(); 3 usages
9
10    // Construtor
11    public Biblioteca(String nomeBiblioteca) 1 usage
12    {
13        this.nomeBiblioteca = nomeBiblioteca;
14        for (Usuario u: usuarios){
15            if (u.getTipo().equals("Professor")){
16                Professor professor = (Professor) u;
17            }
18            Aluno aluno = (Aluno) u;
19        }
20    }
21
22    // Getters e Setters
23    public String getNome(){return this.nomeBiblioteca;} no usages
24    public void setNome(String nome){this.nomeBiblioteca = nome;} no usages
25
26    public ArrayList<Emprestimo> getEmpréstimos(){return this.empréstimos;} no usages
27    public void setEmpréstimos(ArrayList<Emprestimo> empréstimos){this.empréstimos = empréstimos;} no usages
28
29    public ArrayList<Livro> getLivros(){return this.livros;} no usages
30    public void setUsuarios(ArrayList<Livro> livros){this.livros = livros;} no usages
31
32    public ArrayList<Usuario> getUsuarios(){return this.usuarios;} no usages
33    public void setNome(ArrayList<Usuario> usuarios){this.usuarios = usuarios;} no usages
34}
```

```
Sistema-Gerenciador-de-Biblioteca-main Version control Main
Main.java Biblioteca.java Aluno.java x Emprestimo.java Livro.java Usuario.java Professor.java

1 package model;
2 import java.time.LocalDate;
3
4 public class Aluno extends Usuario{ 7 usages
5
6     // Construtor
7     public Aluno(String nome, int ID, String tipo) { 2 usages
8         super(nome, ID, tipo);
9     }
10
11     // Metodo para calcular a data de devolução para o aluno
12     public LocalDate calcularDataFinal(@NotNull LocalDate dataInicial) { 1 usage
13         return dataInicial.plusDays( daysToAdd: 15);
14     }
15 }
16
```

Sistema-Gerenciador-de-Biblioteca-main > src > model > Aluno 16:1 LF UTF-8 4 spaces 02:18

```
Sistema-Gerenciador-de-Biblioteca-main Version control Main
Main.java Biblioteca.java Aluno.java x Emprestimo.java Livro.java Usuario.java Professor.java

1 package model;
2 import java.time.*;import java.time.format.DateTimeFormatter;import java.util.Scanner;
3
4 public class Emprestimo { 8 usages
5     private Livro livro; 4 usages
6     private Usuario usuario; 4 usages
7     private int dia; 2 usages
8     private int mes; 2 usages
9     private int ano; 2 usages
10     public LocalDate dataInicial; 9 usages
11     public LocalDate dataFinal; 7 usages
12
13     // Construtor
14     public Emprestimo(Livro livro, Usuario usuario, int dia, int mes, int ano) { 1 usage
15         Scanner input = new Scanner(System.in);
16
17         this.livro = livro;
18         this.usuario = usuario;
19         this.dataInicial = LocalDate.of(ano, mes, dia);
20         if (dataInicial.isAfter(LocalDate.now())) {
21             while (dataInicial.isAfter(LocalDate.now())) {
22                 System.out.println("A data inserida está inválida. Insira uma data que não seja maior que hoje!");
23                 System.out.println("Insira um novo dia: ");
24                 int diaNovo = input.nextInt();
25                 System.out.println("Insira um novo mes: ");
26                 int mesNovo = input.nextInt();
27                 System.out.println("Insira um novo ano: ");
28                 int anoNovo = input.nextInt();
29                 dataInicial = LocalDate.of(anoNovo, mesNovo, diaNovo);
30             }
31         }
32         this.dataFinal = usuario.calcularDataFinal(dataInicial);
33     }
34 }
```

Sistema-Gerenciador-de-Biblioteca-main > src > model > Emprestimo 12:1 LF UTF-8 4 spaces 02:18

```
1 package model;
2
3 public class Livro {
4     private String titulo;
5     private String autor;
6     private int ano;
7     private int ISBN;
8     private int quantidadeDisponivel;
9
10    // Construtor
11    public Livro(String titulo, String autor, int ano, int ISBN, int quantidadeDisponivel){
12        this.titulo = titulo;
13        this.autor = autor;
14        if (ISBN < 0 || ano < 0){
15            System.out.println("Não é permitido números negativos.");
16            return;
17        } else { this.ISBN = ISBN; this.ano = ano;}
18
19        if(quantidadeDisponivel > 0){
20            this.quantidadeDisponivel = quantidadeDisponivel;
21        } else { System.out.println("Deve existir ao menos 1 livro para ser adicionado a biblioteca futuramente.");}
22    }
23
24    // Getter e setters
25    public String getTitulo(){return this.titulo;}
26    void setTitulo(String titulo){this.titulo = titulo;}
27
28    public String getAutor(){return this.autor;}
29    void setAutor(String autor){this.autor = autor;}
30
31    public int getAno(){return this.ano;}
32    void setAno(int ano){this.ano = ano;}
33 }
```

```
1 package model;
2 import java.time.LocalDate;
3
4 public class Usuario {
5     private String nome;
6     private int ID;
7     private String tipo;
8
9     public Usuario(String nome, int ID, String tipo){
10        this.nome = nome;
11        if (ID < 0){
12            System.out.println("Não é permitido números negativos.");
13        } else { this.ID = ID;}
14        this.tipo = tipo;
15    }
16
17    public String getNome(){return this.nome;}
18    void setNome(String nome){this.nome = nome;}
19
20    public int getID(){return this.ID;}
21    void setID(int ID){this.ID = ID;}
22
23    public String getTipo(){return this.tipo;}
24    void setTipo(String tipo){this.tipo = tipo;}
25
26    void mostrarUsuario(){
27        System.out.println("Nome: " + this.nome);
28        System.out.println("ID de usuário: " + this.ID);
29        System.out.println("O usuário é um: " + this.tipo);
30    }
31
32    public LocalDate calcularDataFinal(@NotNull LocalDate dataInicial){
33 }
```

Sistema-Gerenciador-de-Biblioteca-mainVersion controlMain

Main.javaBiblioteca.javaAluno.javaEmprestimo.javaLivro.javaUsuario.javaProfessor.java

1package model;

2

3import java.time.LocalDate;

4

5

6public class Professor extends Usuario{ 5 usages

7 public Professor(String nome, int ID, String tipo){ 1 usage

8 super(nome, ID, tipo);

9 }

10

11 public LocalDate calcularDataFinal(@NotNull LocalDate dataInicial) { 1 usage

12 return dataInicial.plusDays(daysToAdd: 30);

13 }

14}

15

Sistema-Gerenciador-de-Biblioteca-main > src > model > Professor15:1 LF UTF-8 4 spaces02:19