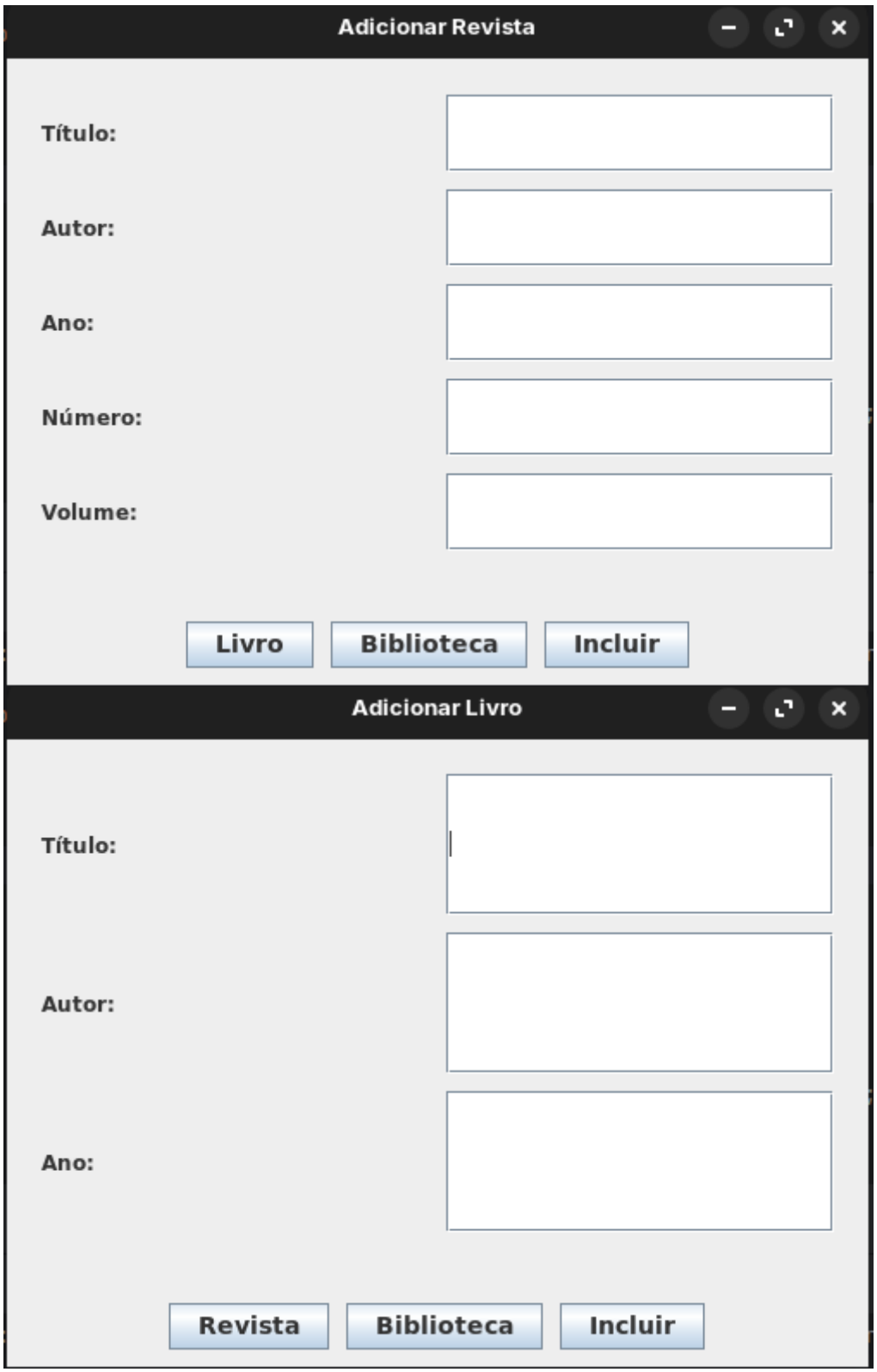
Vitor Ferreira França

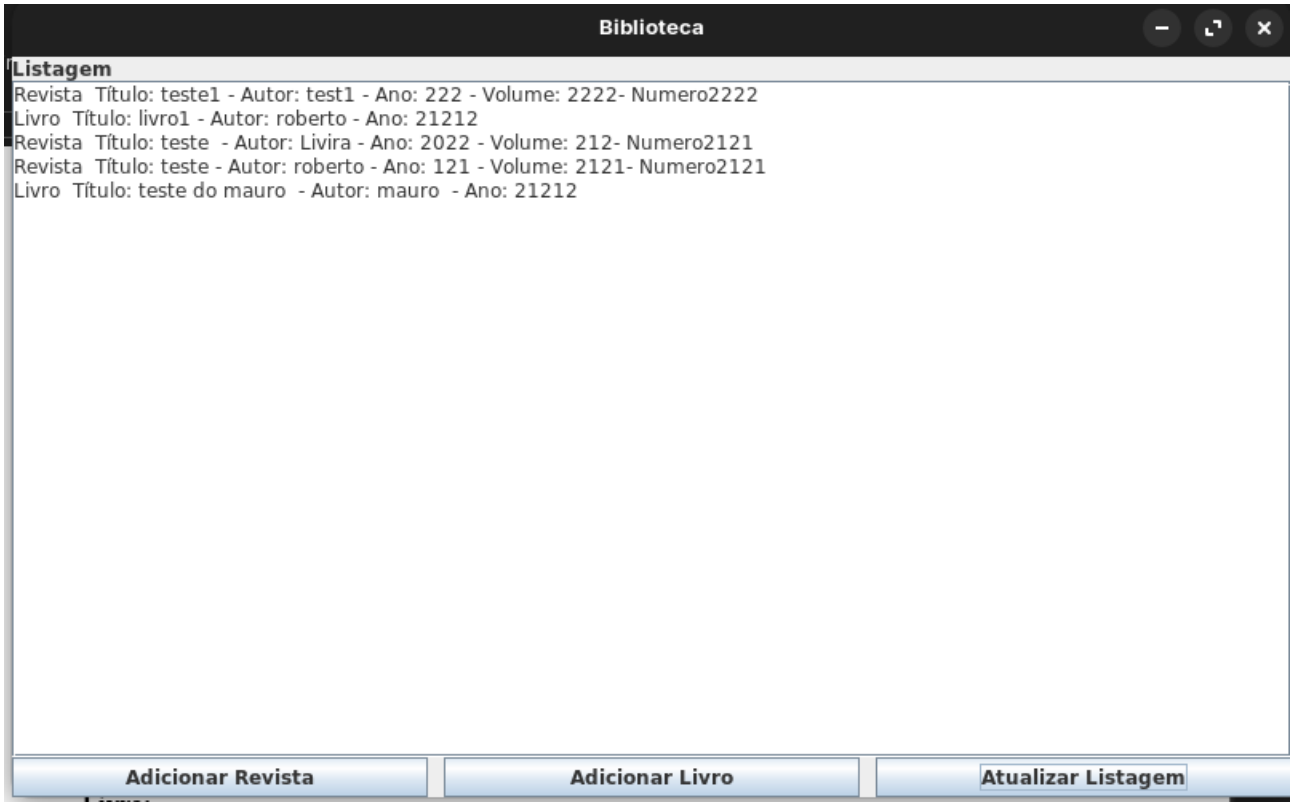
Trabalho 1- Uso de Guis

Link pro Git

-Sumario

1-Telas


Adicionar Revista
Título:
Autor:
Ano:
Número:
Volume:
Livro
Biblioteca
Incluir
Adicionar Livro
Título:
Autor:
Ano:
Revista
Biblioteca
Incluir



2-Estrutura das pastas

As pastas foram organizadas seguindo a estrutura MVC. Na **Model**, estão as classes responsáveis pelos livros e revistas. O **Controller** contém os arquivos responsáveis pela manipulação dos dados. Já a **View** é onde a interface visual da aplicação é construída.

-Estrutura dos Materiais(Livro e Revista)

O livro foi definido como a classe principal, enquanto a revista a estende, pois requer mais dados que não são necessários para o livro. Dessa forma, é possível criar um array contendo ambos.

Livro:

```
package model;

public class Livro {
    private String titulo;
    private String autor;
    private int ano;
```

```
    public String modelo = "Livro";

    public Livro(String titulo,String autor,int ano){
        this.titulo = titulo;
        this.autor = autor;
        this.ano = ano;
    }

    public String getTitulo(){
        return this.titulo;
    }

    public void setTitulo(String titulo){
        this.titulo = titulo;
    }

    public String getAutor(){
        return this.autor;
    }

    public void setAutor(String autor){
        this.autor = autor;
    }

    public int getAno(){
        return this.ano;
    }

    public void setAno(int ano){
        this.ano = ano;
    }

    public String getModelo(){
        return this.modelo;
    }

    public String getAributos(){
        return "Revista{" +
                "titulo='" + this.getTitulo() + '\'' +
                ", autor='" + this.getAutor() + '\'' +
                ", ano="+ this.getAno() +
                '}';
    }

}
```

Revista:

```
package model;



public class Revista extends Livro{
```

```
    private int volume;
    private int numero;
    public Revista(String titulo, String autor, int ano,int volume,int numero) {
        super(titulo, autor, ano); //chama o contrutor da super classe
        this.volume= volume;
        this.numero= numero;
        this.modelo= "Revista";
    }

    public int getVolume(){
        return this.volume;
    }

    public void setVolume(int volume){
        this.volume= volume;
    }

    public int getNumero(){
        return this.numero;
    }

    public void setNumero(int numero){
        this.numero= numero;
    }

    public String getAtributos(){
        return "Revista{" +
                "titulo='" + this.getTitulo() + '\'' +
                ", autor='" + this.getAutor() + '\'' +
                ", ano="+ this.getAno() +
                ", volume=" + volume +
                ", numero=" + numero +
                '}';
    }
}
```

3-Estrutura das View’s

Cada página possui sua própria View: `Main`, `RevistaView` e `LivroView`. As Views de revista e livro são responsáveis por adicionar itens à biblioteca principal, gerenciada pela `MainView`, onde o array com todos os itens é armazenado.

Cada View recebe uma referência ao array compartilhado, garantindo que todos os itens sejam armazenados corretamente. Ao abrir uma nova tela, qualquer outra tela aberta é fechada automaticamente, exceto a `Main`, que apenas oculta sua visualização sem ser fechada.

Main:

```
package view;

import model.*;
import java.util.ArrayList;
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class Main {
    public static void main(String[] args) {

        ArrayList<Livro> listaMateriais = new ArrayList<>();

        JFrame janelaMain = new JFrame();
        JLabel Listagem = new JLabel("Listagem");
        janelaMain.setSize(800, 500);
        janelaMain.setTitle("Biblioteca");
        janelaMain.setLocationRelativeTo(null);
        janelaMain.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        janelaMain.setLayout(new BorderLayout());

                // Painel principal
        JPanel painelBotoes = new JPanel();
        painelBotoes.setLayout(new GridLayout(1, 2, 10, 10));

        JButton adicionarRevista = new JButton("Adicionar Revista");
        adicionarRevista.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
                janelaMain.setVisible(false);
                new RevistaView(listaMateriais, janelaMain);
            }
        });

        JButton adicionarLivro = new JButton("Adicionar Livro");
        adicionarLivro.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
                janelaMain.setVisible(false);
                new LivroView(listaMateriais, janelaMain);
            }
        });

        painelBotoes.add(adicionarRevista);
        painelBotoes.add(adicionarLivro);

        //  Painel de Listagem
        JPanel painelListagem = new JPanel(new BorderLayout());

        JTextArea txtLista = new JTextArea();
        txtLista.setEditable(false);
```

```
        JScrollPane scrollPane = new JScrollPane(txtLista);

        JButton btnAtualizar = new JButton("Atualizar Listagem");
        btnAtualizar.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
                txtLista.setText(""); // Limpa antes de listar
                for (Livro material : listaMateriais) {
                    if (material instanceof Revista) {
                        Revista revista = (Revista) material; // Faz o cast para
Revista
                        txtLista.append(
                                revista.getModelo()+ "  "+
                                "Título: " + revista.getTitulo() +
                                " - Autor: " + revista.getAutor() +
                                " - Ano: " + revista.getAno() +
                                " - Volume: " + revista.getVolume() +
                                "- Numero" +revista.getNumero()+"\n"
                                );
                    } else {
                        txtLista.append(
                                material.getModelo()+ "  "+
                                "Título: " + material.getTitulo() +
                                " - Autor: " + material.getAutor() +
                                " - Ano: " + material.getAno() + "\n"
                                );
                    }
                }
            }
        });

        painelListagem.add(scrollPane, BorderLayout.CENTER);
        painelListagem.add(Listagem, BorderLayout.NORTH);

        painelBotoes.add(btnAtualizar);

        // Adiciona os painéis na janela principal
        janelaMain.add(painelBotoes, BorderLayout.SOUTH);
        janelaMain.add(painelListagem, BorderLayout.CENTER);

        janelaMain.setVisible(true);
    }
}
```

RevistaView:

```
package view;


import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
```

```java
import java.awt.event.ActionListener;
import java.util.ArrayList;

import controller.RevistaController;
import model.Livro;
import model.Revista;


public class RevistaView{
    public RevistaView(ArrayList<Livro> listaObjetos, JFrame JanelaMain){
//construtor

        // Criando a janela principal
        JFrame jframe = new JFrame("Adicionar Revista");
        jframe.setSize(500, 400);
        jframe.setLocationRelativeTo(null);
        jframe.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        jframe.setLayout(new BorderLayout(10, 10));

        // Painel para os campos de entrada
        JPanel painelCampos = new JPanel(new GridLayout(5, 2, 10, 10));
        painelCampos.setBorder(BorderFactory.createEmptyBorder(20, 20, 20, 20));

        JLabel tituloLabel = new JLabel("Título:");
        JTextField tituloField = new JTextField();

        JLabel autorLabel = new JLabel("Autor:");
        JTextField autorField = new JTextField();

        JLabel anoLabel = new JLabel("Ano:");
        JTextField anoField = new JTextField();

        JLabel numeroLabel = new JLabel("Número:");
        JTextField numeroField = new JTextField();

        JLabel volumeLabel = new JLabel("Volume:");
        JTextField volumeField = new JTextField();

        // Adicionando os componentes ao painel
        painelCampos.add(tituloLabel);
        painelCampos.add(tituloField);
        painelCampos.add(autorLabel);
        painelCampos.add(autorField);
        painelCampos.add(anoLabel);
        painelCampos.add(anoField);
        painelCampos.add(numeroLabel);
        painelCampos.add(numeroField);
        painelCampos.add(volumeLabel);
        painelCampos.add(volumeField);

        // Criando os botões
        JButton adicionarBtn = new JButton("Incluir");
        JButton bibliotecaBtn = new JButton("Biblioteca");
        JButton abrirLivroBtn = new JButton("Livro");
```

```
        // Ajustando a fonte dos botões
        Font botaoFonte = new Font("Arial", Font.BOLD, 14);
        adicionarBtn.setFont(botaoFonte);
        bibliotecaBtn.setFont(botaoFonte);
        abrirLivroBtn.setFont(botaoFonte);

        // Definição do controlador
        RevistaController controller = new RevistaController();

        // Ação do botão Adicionar
        adicionarBtn.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                try {
                    String titulo = tituloField.getText();
                    String organizacao = autorField.getText();
                    int ano = Integer.parseInt(anoField.getText());
                    int numero = Integer.parseInt(numeroField.getText());
                    int volume = Integer.parseInt(volumeField.getText());

                    // Limpar os campos após adicionar
                    tituloField.setText("");
                    autorField.setText("");
                    anoField.setText("");
                    numeroField.setText("");
                    volumeField.setText("");

                    controller.adicionaRevista(titulo, organizacao, ano, volume,
numero, listaObjetos);

                    JOptionPane.showMessageDialog(jframe, "Revista adicionada
com sucesso!", "Sucesso", JOptionPane.INFORMATION_MESSAGE);

                } catch (NumberFormatException ex) {
                    JOptionPane.showMessageDialog(jframe, "Erro: Ano, Número e
Volume precisam ser números.", "Erro", JOptionPane.ERROR_MESSAGE);
                }
            }
        });

        // Ação do botão Biblioteca
        bibliotecaBtn.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
                jframe.dispose();
                JanelaMain.setVisible(true);
            }
        });

        // Ação do botão Abrir Livro
        abrirLivroBtn.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
                jframe.dispose();
                new LivroView(listaObjetos, JanelaMain);
            }
```

```
        });

        // Painel de botões
        JPanel painelBotoes = new JPanel(new FlowLayout(FlowLayout.CENTER, 10,
10));
        painelBotoes.add(abrirLivroBtn);
        painelBotoes.add(bibliotecaBtn);
        painelBotoes.add(adicionarBtn);

        // Adicionando os componentes à janela
        jframe.add(painelCampos, BorderLayout.CENTER);
        jframe.add(painelBotoes, BorderLayout.SOUTH);

        jframe.setVisible(true);
    }
```

LivroView:

```
package view;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;

import controller.LivroController;
import model.*;
public class LivroView {
    public LivroView(ArrayList<Livro> listaLivro, JFrame janelaMain) {
        // Criando a janela principal
        JFrame jframe = new JFrame("Adicionar Livro");
        jframe.setSize(500, 400);
        jframe.setLocationRelativeTo(null);
        jframe.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        jframe.setLayout(new BorderLayout(10, 10));

        // Painel para os campos de entrada
        JPanel painelCampos = new JPanel(new GridLayout(3, 2, 10, 10));
        painelCampos.setBorder(BorderFactory.createEmptyBorder(20, 20, 20, 20));

        JLabel tituloLabel = new JLabel("Título:");
        JTextField tituloField = new JTextField();

        JLabel autorLabel = new JLabel("Autor:");
        JTextField autorField = new JTextField();

        JLabel anoLabel = new JLabel("Ano:");
        JTextField anoField = new JTextField();

        // Adicionando componentes ao painel
        painelCampos.add(tituloLabel);
        painelCampos.add(tituloField);
```

```
        painelCampos.add(autorLabel);
        painelCampos.add(autorField);
        painelCampos.add(anoLabel);
        painelCampos.add(anoField);

        // Criando os botões
        JButton adicionarBtn = new JButton("Incluir");
        JButton bibliotecaBtn = new JButton("Biblioteca");
        JButton abrirRevistaBtn = new JButton("Revista");

        // Ajustando a fonte dos botões
        Font botaoFonte = new Font("Arial", Font.BOLD, 14);
        adicionarBtn.setFont(botaoFonte);
        bibliotecaBtn.setFont(botaoFonte);
        abrirRevistaBtn.setFont(botaoFonte);

        // Definição do controlador
        LivroController controller = new LivroController();

        // Ação do botão Adicionar
        adicionarBtn.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                try {
                    String titulo = tituloField.getText();
                    String autor = autorField.getText();
                    int ano = Integer.parseInt(anoField.getText());

                    // Limpar os campos após adicionar
                    tituloField.setText("");
                    autorField.setText("");
                    anoField.setText("");

                    controller.AdicionaLivro(titulo, autor, ano, listaLivro);

                    JOptionPane.showMessageDialog(jframe, "Livro adicionado com
sucesso!", "Sucesso", JOptionPane.INFORMATION_MESSAGE);

                } catch (NumberFormatException ex) {
                    JOptionPane.showMessageDialog(jframe, "Erro: Ano deve ser um
número.", "Erro", JOptionPane.ERROR_MESSAGE);
                }
            }
        });

        // Ação do botão Biblioteca
        bibliotecaBtn.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
                jframe.dispose();
                janelaMain.setVisible(true);
            }
        });

        // Ação do botão Abrir Revista
        abrirRevistaBtn.addActionListener(new ActionListener() {
```

```
            @Override
            public void actionPerformed(ActionEvent actionEvent) {
                jframe.dispose();
                new RevistaView(listaLivro, janelaMain);
            }
        });

        // Painel de botões
        JPanel painelBotoes = new JPanel(new FlowLayout(FlowLayout.CENTER, 10,
10));
        painelBotoes.add(abrirRevistaBtn);
        painelBotoes.add(bibliotecaBtn);
        painelBotoes.add(adicionarBtn);

        // Adicionando os componentes à janela
        jframe.add(painelCampos, BorderLayout.CENTER);
        jframe.add(painelBotoes, BorderLayout.SOUTH);

        jframe.setVisible(true);
    }
}
```

4-Controllers

Tanto o `Livro` quanto a `Revista` receberam um controlador, responsável pela manipulação dos dados e funcionalidades específicas de cada um. É nele que serão implementadas as operações e demais funcionalidades relacionadas a cada item.

RevistaController:

```
package controller;
import model.*;

import java.awt.event.ActionListener;
import java.util.ArrayList;


public class RevistaController
{
    public void adicionaRevista(String titulo, String organizacao, int ano, int
volume, int numero, ArrayList<Livro> lista) {
        Revista revista = new Revista(titulo, organizacao, ano, volume, numero);
        lista.add(revista);
    }
}
```

LivroController:

```
package controller;

import java.util.ArrayList;
import  model.*;

public class LivroController {
    public void AdicionaLivro(String titulo,String autor,int ano,
ArrayList<Livro> ListaLivro){
        Livro revista = new Livro(titulo,autor,ano);
        ListaLivro.add(revista);
    }
}
```

5-Erros cometidos

1-Para a troca de páginas, poderia ter sido criada uma única função reutilizável, evitando a duplicação de código e tornando a implementação mais eficiente.

```
abrirLivroBtn.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent actionEvent) {
        jframe.dispose();
        new LivroView(listaObjetos, JanelaMain);
    }
});
```

e

```
abrirRevistaBtn.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent actionEvent) {
        jframe.dispose();
        new RevistaView(listaLivro, janelaMain);
    }
});
```

2 – A View acabou incorporando funções do Controller, como as ações dos botões, pois foi minha primeira vez utilizando “`ActionListener()`” e tive dificuldades em chamar funções de outro arquivo.