
Engenharia De Computação
POCO4A - Programação Orientada a Objetos

Projeto Final Programação Orientada a Objetos

DEIVID DA SILVA GALVÃO
JOÃO VITOR NAKAHODO YOSHIDA

DEIVID DA SILVA GALVÃO
JOÃO VITOR NAKAHODO YOSHIDA

Projeto Final Programação Orientada a Objetos

Relatório Técnico do Trabalho Disciplinar apresentado como requisito parcial à obtenção de nota na disciplina de Programação orientada a objetos do Curso Superior de Engenharia de Computação da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Lucio Agostinho Rocha

SUMÁRIO

1. INTRODUÇÃO.....	4
2. MODELAGEM DO PROGRAMA.....	4
3. IMPLEMENTAÇÃO.....	6
4. CONCLUSÃO.....	12
5. REFERÊNCIAS.....	13

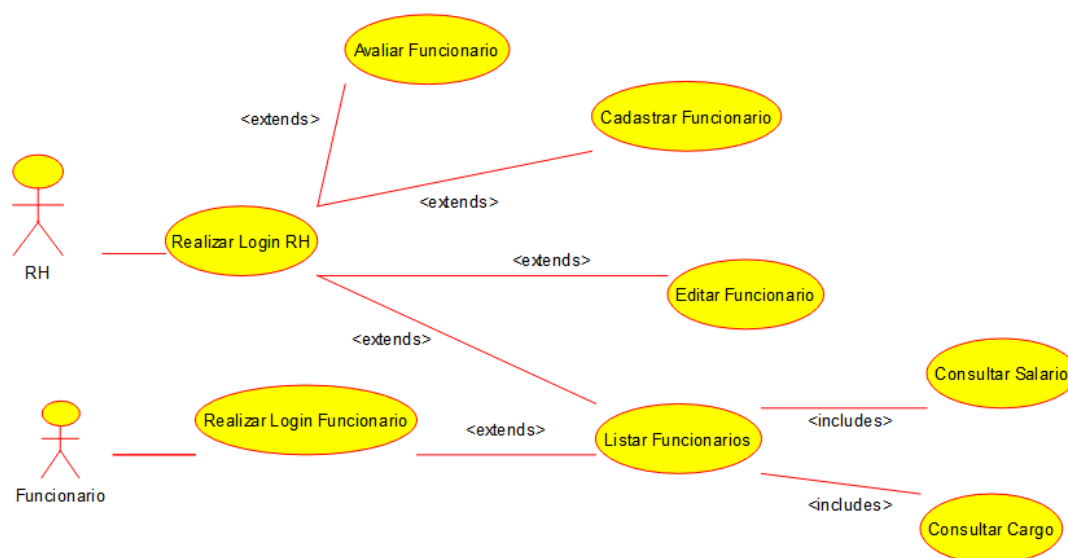
1. INTRODUÇÃO

A Programação Orientada a Objetos (POO) é uma abordagem essencial no desenvolvimento de software, que permite a criação de sistemas complexos e modulares por meio da organização em objetos interativos. Ao longo do semestre, adquirimos conhecimentos teóricos e práticos sobre os princípios e técnicas dessa metodologia. Este relatório apresenta os detalhes e os resultados do nosso projeto final de Programação Orientada a Objetos, cujo objetivo foi desenvolver um programa para gerenciar o departamento de Recursos Humanos (RH) da prefeitura. Foi utilizada a linguagem de programação Java e exploramos diversas funcionalidades para criar uma solução eficiente e completa.

2. MODELAGEM DO PROGRAMA

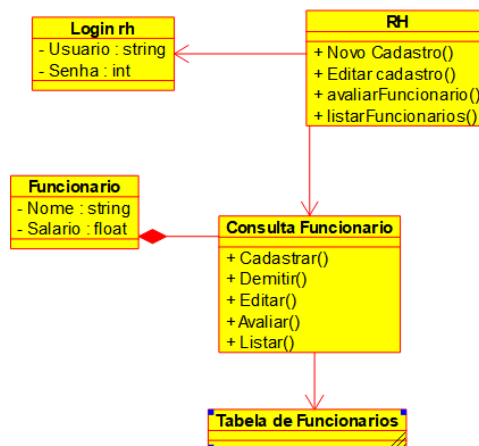
O projeto tem como objetivo a criação de um software para gerenciar o sistema de RH de uma prefeitura inicialmente foi realizada a modelagem do projeto por meio do software “Umbrello”, com o intuito de evidenciar os casos de uso que seriam implementados no programa juntamente com os atores como mostra a figura abaixo:

Figura 1 – Diagrama de Casos de Uso



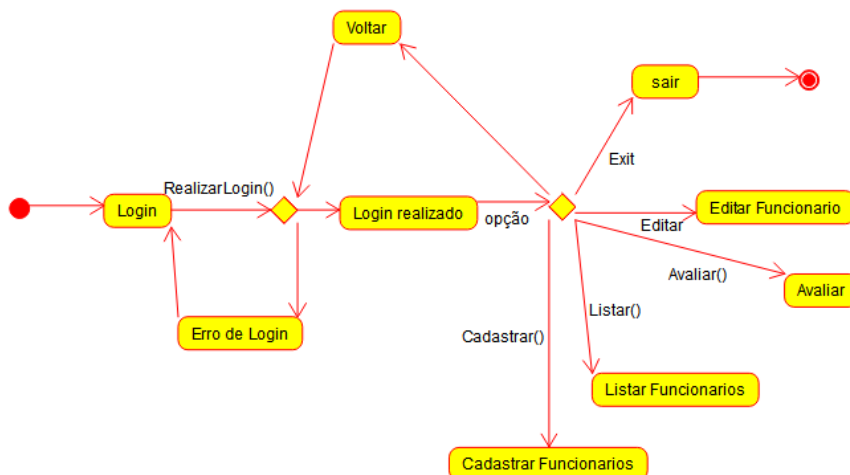
Logo em seguida foi desenvolvido o diagrama de classes (Figura 2) onde todas as classes e seus respectivos métodos são evidenciados, o diagrama de atividades (Figura 3), o diagrama e o diagrama de estados (Figura 4), porém com o andamento do desenvolvimento do código foram realizadas algumas alterações.

Figura 2 – Diagrama de Classes.



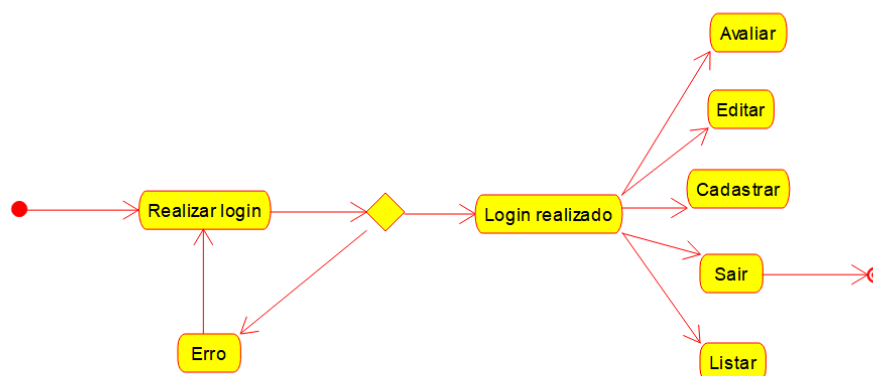
Fonte: Autoria Própria (2023).

Figura 3 – Diagrama de Atividades.



Fonte: Autoria Própria (2023).

Figura 4 – Diagrama de Estados.



Fonte: Autoria Própria (2023).

3. IMPLEMENTAÇÃO

Para a implementação do projeto foi utilizada a linguagem de programação Java , por meio da IDE netbeans.

Inicialmente foi feita a tela de login onde funcionários “comuns” e usuários do RH podem efetuar login no sistema se efetivado cada tipo de usuário vai para a determinada janela correspondente como no exemplo abaixo:

Imagem 1 – Tela de Login.

Fonte: Autoria Própria (2023).

Para o caso do login como “RH” foi definido um usuário de login padrão onde o campo usuário deve ser preenchido como admin e a senha deve ser preenchida como “root123” como mostra o código abaixo:

Imagem 2 – Código da Classe Principal

```
public class Principal {
    public static ArrayList<ILogin> listaLogin = new ArrayList<>();
    public static ArrayList<IDados> listaDados = new ArrayList<>();

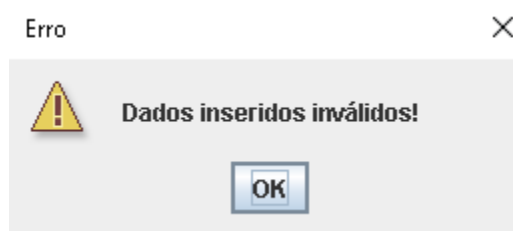
    public Principal() {
        ILogin login = new RH("admin", "root123");
        login.cadastrar(login);
        Login inicio = Login.iniciar();
        inicio.setVisible(true);
    }

    public static void main(String[] args) {
        new Principal();
    }
}
```

Fonte: Autoria Própria (2023).

Caso o usuário insira dados incorretos e que não estão na “base de dados” do sistema é disparada uma janela de erro e o login não é realizado.

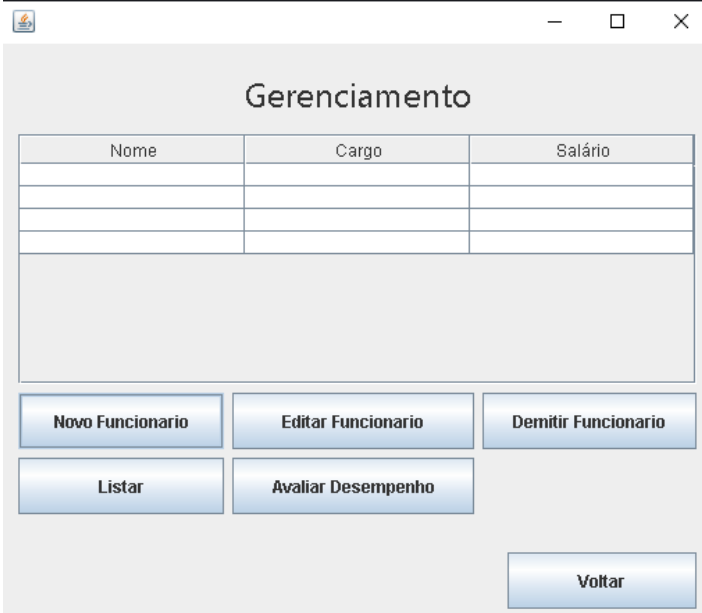
Imagem 3 – Janela de erro de login



Fonte: Autoria Própria (2023).

Se o login for efetivado com sucesso pelo administrador ele será encaminhado para a janela de RH

Imagem 4 – Janela do RH



Nome	Cargo	Salário

Novo Funcionario

Editar Funcionario

Demitir Funcionario

Listar

Avaliar Desempenho

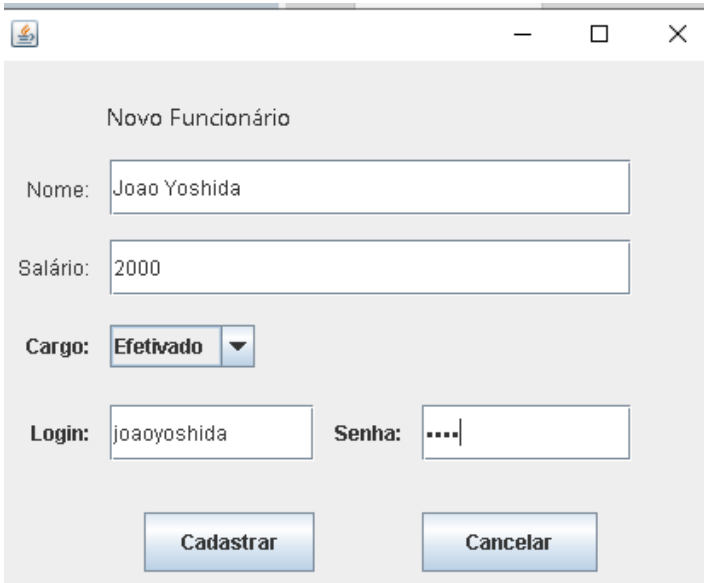
Voltar

Fonte: Autoria Própria (2023).

Na janela de RH são disponibilizadas as opções de criar um novo funcionário, editar um funcionário já existente, demitir funcionário, listar todos os funcionários já cadastrados e também a opção de avaliar o desempenho de um funcionário que já possua cadastro.

Cada opção leva o usuário a uma determinada janela, por exemplo se a intenção for cadastrar um novo funcionário será exibida uma janela conforme a imagem abaixo:

Imagem 5 – Janela de cadastro de funcionário



Novo Funcionario

Nome: Joao Yoshida

Salário: 2000

Cargo: Efetivado ▼

Login: joaoyoshida Senha:

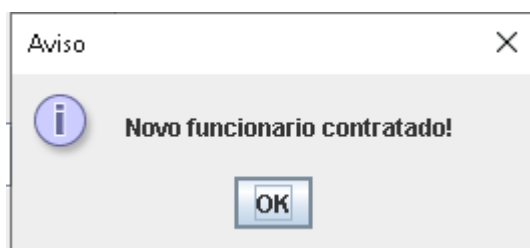
Cadastrar

Cancelar

Fonte: Autoria Própria (2023).

Caso o novo funcionário e seus dados sejam escritos de maneira válida o novo funcionário é cadastrado e é exibida uma janela de aviso confirmando que o cadastro foi bem sucedido como na imagem abaixo:

Imagem 6 – Janela de confirmação de novo cadastro.



Fonte: Autoria Própria (2023).

Porém se forem inseridos dados inválidos no campo de salário como uma letra o software por meio do tratamento de exceções ,try catch, não deixa que o cadastro seja efetivado.

Imagem 7 – Exemplo de Tratamento de exceção no código

```

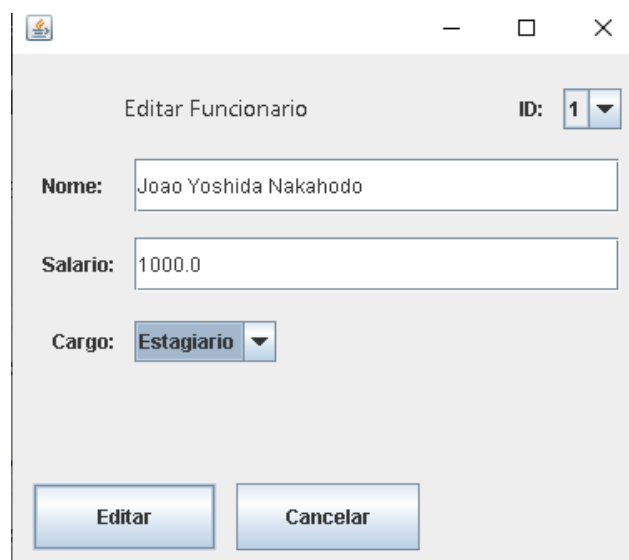
193 private void txtSalarioFuncionarioActionPerformed(java.awt.event.ActionEvent evt) {GEN-FIRST:event_txtSalarioFuncionarioActionPerformed
194     try {
195         Float salario = Float.parseFloat(txtSalarioFuncionario.getText());
196     } catch (NumberFormatException e) {
197         // Capturar a exceção caso o valor não seja um número inteiro
198         JOptionPane.showMessageDialog(null, "Salário digitado inválido", "Erro", JOptionPane.ERROR_MESSAGE);
199     }
200 } //GEN-LAST:event_txtSalarioFuncionarioActionPerformed

```

Fonte: Autoria Própria (2023).

Caso o usuário do RH escolha editar um funcionário cadastrado ele é direcionado para a janela da imagem a seguir:

Imagem 8 – Janela de editar funcionário



Fonte: Autoria Própria (2023).

Onde poderá ser alterado o nome, o cargo e o salário de determinado funcionário, caso a alteração seja concluída com sucesso é exibida para o usuário uma mensagem de confirmação conforme a imagem abaixo:

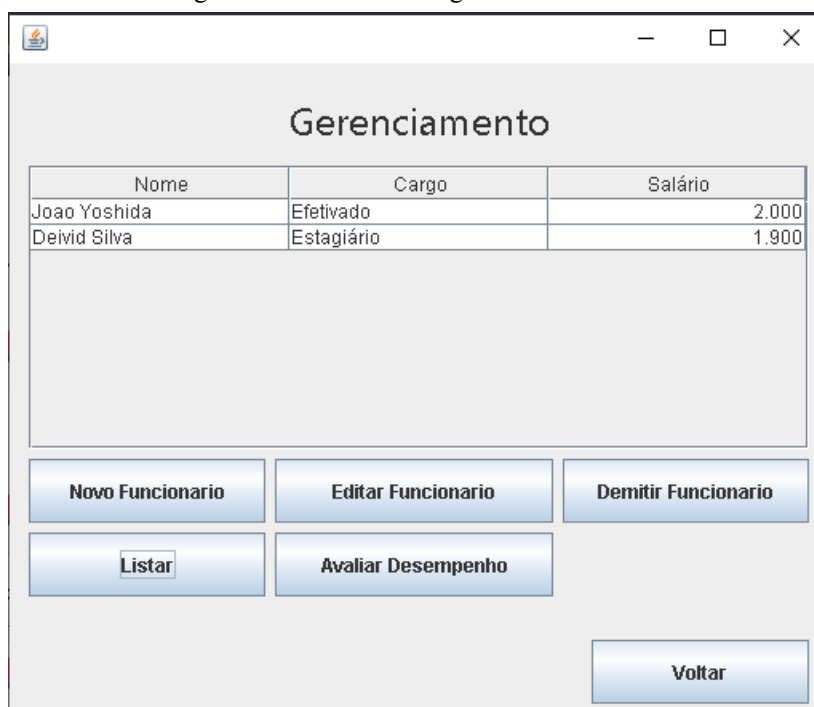
Imagem 9 – Janela de confirmação de edição de funcionário



Fonte: Autoria Própria (2023).

Se RH desejar listar todos os funcionários que possuem cadastro ativo será exibida uma outra janela conforme a imagem abaixo:

Imagem 10 – Janela de listagem dos funcionários



Fonte: Autoria Própria (2023).

Caso a opção escolhida pelo usuário seja “demitir funcionário” ele será direcionado para a tela de demissão onde é possível desvincular um funcionário já cadastrado, e também é exibida uma janela de confirmação caso a demissão se conclua.

Imagem 11 – Janela de demissão dos funcionários



Fonte: Autoria Própria (2023).

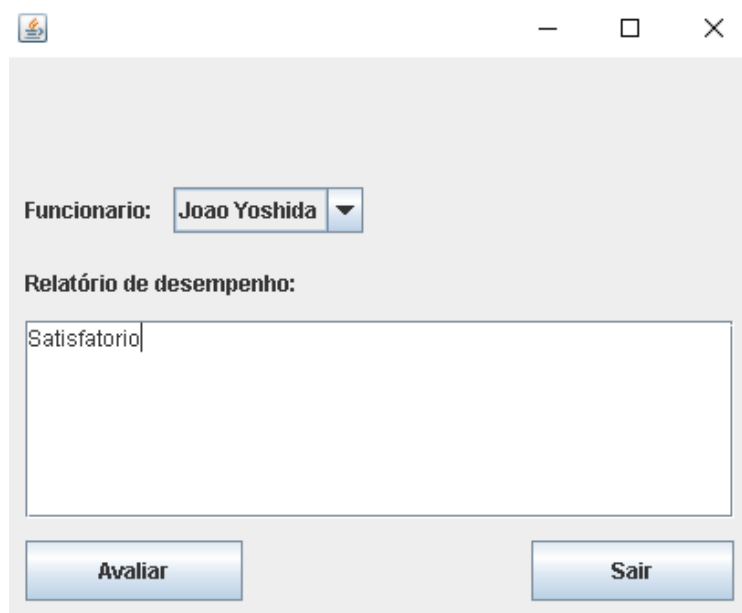
Imagem 12 – Janela de confirmação de demissão de funcionário



Fonte: Autoria Própria (2023).

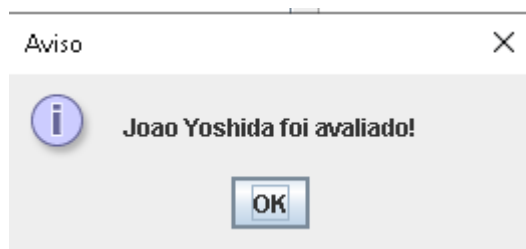
Por fim, caso a opção escolhida pelo usuário seja “Avaliar Desempenho” ele será direcionado para a tela onde será possível escrever um relatório de desempenho para o funcionário escolhido, caso a avaliação seja realizada com sucesso uma mensagem de confirmação é disparada.

Imagem 13 – Janela de Avaliação dos funcionários



Fonte: Autoria Própria (2023).

Imagem 14 – Janela de confirmação de avaliação



Fonte: Autoria Própria (2023).

O padrão de projeto utilizado no código foi o Singleton. O Singleton é um padrão de criação que garante a existência de apenas uma instância de uma determinada classe durante a execução do programa. Ele é comumente usado quando é necessário compartilhar um único objeto entre vários componentes do sistema. No trecho de código da imagem a baixo a implementação do Singleton pode ser identificada:

Imagem 15 – Exemplo de padrão Singleton no código

```
public class TelaRH extends javax.swing.JFrame {
    private static TelaRH janela;

    /**
     * Creates new form TelaRH
     */
    private TelaRH() {
        initComponents();
    }

    public static TelaRH iniciar() {
        if (janela == null) {
            janela = new TelaRH();
        }
        return janela;
    }
}
```

Fonte: Autoria Própria (2023).

4. CONCLUSÃO

Neste relatório técnico, apresentamos os detalhes e resultados do nosso projeto final de Programação Orientada a Objetos. Nosso objetivo foi desenvolver um programa para gerenciar o departamento de Recursos Humanos (RH) de uma prefeitura, utilizando a linguagem de programação Java.

Este trabalho nos permitiu aprimorar nossas habilidades em programação, modelagem de sistemas e trabalho em equipe. Através do desenvolvimento desse projeto, fomos capazes de aplicar os conhecimentos adquiridos ao longo do curso, enfrentar desafios e buscar soluções criativas.

Em resumo, o projeto final de Programação Orientada a Objetos foi uma experiência enriquecedora que nos proporcionou a oportunidade de consolidar nosso aprendizado e demonstrar nossas habilidades no desenvolvimento de software orientado a objetos.

5. REFERÊNCIAS

Link do código completo:

[Programacao-orientada-a-objetos/Principal-20230629T210201Z-001/Principal/src/main/java at main · IF-DeividSilva/Programacao-orientada-a-objetos · GitHub](#)

POCO4A - 2s2023. [S. l.], 2023. Disponível em:

<https://classroom.google.com/u/1/c/NTI2MTA1NTc5NTI2>. Acesso em: 29 junho 2023