

**UNIVERSIDADE PAULISTA**

**ERYCK DOS SANTOS FERREIRA  
HENRIQUE SILVA CHAGAS  
KAYKI DE SOUZA MOLINA  
LUCAS FRANCISCO DE OLIVEIRA**

**PROJETO INTEGRADO MULTIDISCIPLINAR III**

**São Paulo  
2023**

**ERYCK DOS SANTOS FERREIRA  
HENRIQUE SILVA CHAGAS  
KAYKI DE SOUZA MOLINA  
LUCAS FRANCISCO DE OLIVEIRA**

**PROJETO INTEGRADO MULTIDISCIPLINAR III**

Projeto Integrado Multidisciplinar para obtenção  
do título de Tecnólogo em Análise e Desenvolvi-  
mento de Sistemas apresentado à Universidade  
Paulista – UNIP.

Orientador: Prof. Fabio Assis  
Coorientador: Prof. Hugo Fernandes

**São Paulo  
2023**

CIP - Catalogação na Publicação

Projeto Integrado Multidisciplinar III / Henrique Chagas...[et al.]. -  
2023.

29 f. : il. color

Trabalho de Conclusão de Curso (Graduação) apresentado ao Instituto  
de Ciência Exatas e Tecnologia da Universidade Paulista, São Paulo,  
2023.

Área de Concentração: Análise e Desenvolvimento de Sistemas.

Orientador: Prof. Me. Fabio Assis.

Coorientador: Prof. Esp.Hugo Fernandes.

1. Automação. 2. Folha de Pagamento. 3. Aplicação Web. 4. Aplicação  
Desktop. 5. Desenvolvimento de Software. I. Chagas, Henrique. II. Assis,  
Fabio (orientador). III. Fernandes, Hugo (coorientador).

**ERYCK DOS SANTOS FERREIRA**

**KAYKI DE SOUZA MOLINA**

**HENRIQUE SILVA CHAGAS**

**LUCAS FRANCISCO DE OLIVEIRA**

**PROJETO INTEGRADO MULTIDISCIPLINAR III**

Projeto Integrado Multidisciplinar para  
obtenção do título de Tecnólogo em Análise e  
Desenvolvimento de Sistemas apresentado à  
Universidade Paulista – UNIP.

Aprovado em: \_\_\_\_ / \_\_\_\_ / \_\_\_\_.

**BANCA EXAMINADORA**

\_\_\_\_ / \_\_\_\_ / \_\_\_\_

Prof. Fabio Assis (Orientador)

\_\_\_\_ / \_\_\_\_ / \_\_\_\_

Prof. Hugo Fernandes (Coorientador)

Universidade Paulista - UNIP

## **Resumo**

A folha de pagamento é uma parte fundamental no contrato entre todo funcionário e empresa, um documento onde apresenta clareza e transparência de como a remuneração do empregado é distribuído, entre o seu valor bruto, os descontos e o salário líquido ao final. Com a constante evolução tecnológica esse processo uma vez manual eventualmente se tornou automatizado, e atualmente esse processo é mais relacionado a própria revolução digital. Devido a essa constante modernização, o desenvolvimento de Softwares para a realização dessa tarefa se tornou algo importante, e neste documento se busca detalhar o desenvolvimento de uma versão do mesmo, onde são apresentadas diversas funcionalidades relacionados a essa tarefa, além de seu uso em diversas plataformas. Ao final deste desenvolvimento, o ciclo de criação e implementação desse Software, além das metodologias usadas no mesmo estará declarada, servindo como um manual de uso do mesmo.

**Palavras-chave:** Automação; Folha de Pagamento; Desenvolvimento de Software; Implementação; Manual do Usuário.

## **Abstract**

The payroll is a fundamental part of the contract between every employee and company, a document that provides clarity and transparency as to how the employee's pay is distributed, between the gross amount, the deductions and the net salary at the end. With constant technological evolution, this once manual process eventually became automated, and today this process is more related to the digital revolution itself. Due to this constant modernization, the development of software to carry out this task has become important, and this document seeks to detail the development of a version of it, where various functionalities related to this task are presented, in addition to its use on various platforms. At the end of this development, the cycle of creation and implementation of this software, as well as the methodologies used in it, will be declared, serving as a manual for its use.

**Keywords:** Automation; Payroll; Software Development; Implementation; User Manual.

## Lista de ilustrações

Figura 1 – Exemplo de Modelo Incremental . . . . .	18
Figura 2 – Caso de Uso demonstrando as funcionalidades da aplicação . . . . .	20
Figura 3 – Tela de Login . . . . .	21
Figura 4 – Menu Principal . . . . .	21
Figura 5 – Relatório de Funcionários . . . . .	22
Figura 6 – Cadastro de Funcionário . . . . .	22
Figura 7 – Edição de Cadastro . . . . .	23
Figura 8 – Exclusão de Cadastro . . . . .	23
Figura 9 – Consulta de Cadastro . . . . .	24
Figura 10 – Emissão de Folha de Pagamento . . . . .	24
Figura 11 – Diagrama DER do Banco de Dados . . . . .	25
Figura 12 – Diagrama de Classes . . . . .	26
Figura 13 – Tabela com a representação das informações dos funcionários . . . . .	26
Figura 14 – Tabela representando as credenciais de Login do Sistema . . . . .	26
Figura 15 – Tela de Login da Aplicação Web . . . . .	27
Figura 16 – Menu Principal da Aplicação Web . . . . .	27
Figura 17 – Cadastro de Funcionário na Aplicação Web . . . . .	28
Figura 18 – Emissão de Folha de Pagamentos na Aplicação Web . . . . .	28
Figura 19 – Layout do aplicativo . . . . .	29
Figura 20 – Cálculo sem Vale-Transporte . . . . .	30
Figura 21 – Cálculo com Vale-Transporte . . . . .	32
Figura 22 – Mensagem de erro . . . . .	33
Figura 23 – Anexo 1 - Código fonte da página de Login do App Web . . . . .	37
Figura 24 – Anexo 2 - Código fonte do Menu Principal do App Web . . . . .	38
Figura 25 – Anexo 3 - Código fonte da página de cadastro do App Web [1/4] . . . . .	39
Figura 26 – Anexo 4 - Código fonte da página de cadastro do App Web [2/4] . . . . .	40
Figura 27 – Anexo 5 - Código fonte da página de cadastro do App Web [3/4] . . . . .	41
Figura 28 – Anexo 6 - Código fonte da página de cadastro do App Web [4/4] . . . . .	42
Figura 29 – Anexo 7 - Código fonte da página de geração de folha de pagamentos [1/2] .	43
Figura 30 – Anexo 8 - Código fonte da página de geração de folha de pagamentos [2/2] .	44
Figura 31 – Anexo 9 - Representação em Código XML do Layout do App Android [1/6]	45
Figura 32 – Anexo 10 - Representação em Código XML do Layout do App Android [2/6]	46
Figura 33 – Anexo 11 - Representação em Código XML do Layout do App Android [3/6]	47
Figura 34 – Anexo 12 - Representação em Código XML do Layout do App Android [4/6]	48
Figura 35 – Anexo 13 - Representação em Código XML do Layout do App Android [5/6]	49
Figura 36 – Anexo 14 - Representação em Código XML do Layout do App Android [6/6]	50
Figura 37 – Anexo 15 - Código de execução em Java do App Android [1/3] . . . . .	51

Figura 38 – Anexo 16 - Código de execução em Java do App Android [2/3] . . . . .	52
Figura 39 – Anexo 17 - Código de execução em Java do App Android [3/3] . . . . .	53
Figura 40 – Anexo 18 - Objeto comunicador com o Banco de Dados [1/8] . . . . .	54
Figura 41 – Anexo 19 - Objeto comunicador com o Banco de Dados [2/8] . . . . .	54
Figura 42 – Anexo 20 - Objeto comunicador com o Banco de Dados [3/8] . . . . .	55
Figura 43 – Anexo 21 - Objeto comunicador com o Banco de Dados [4/8] . . . . .	55
Figura 44 – Anexo 22 - Objeto comunicador com o Banco de Dados [5/8] . . . . .	56
Figura 45 – Anexo 23 - Objeto comunicador com o Banco de Dados [6/8] . . . . .	56
Figura 46 – Anexo 24 - Objeto comunicador com o Banco de Dados [7/8] . . . . .	57
Figura 47 – Anexo 25 - Objeto comunicador com o Banco de Dados [8/8] . . . . .	57
Figura 48 – Anexo 26 - Objeto relacionado a função de consulta de cadastro [1/2] . . . .	58
Figura 49 – Anexo 27 - Objeto relacionado a função de consulta de cadastros [2/2] . . .	58
Figura 50 – Anexo 28 - Objeto relacionado a função de edição de cadastros [1/2] . . . .	59
Figura 51 – Anexo 29 - Objeto relacionado a função de edição de cadastros [2/2] . . . .	59
Figura 52 – Anexo 30 - Objeto relacionado a função de emitir a folha de pagamento [1/3]	60
Figura 53 – Anexo 31 - Objeto relacionado a função de emitir a folha de pagamento [2/3]	60
Figura 54 – Anexo 32 - Objeto relacionado a função de emitir a folha de pagamento [3/3]	61
Figura 55 – Anexo 33 - Objeto relacionado a função de exclusão de cadastro [1/2] . . . .	61
Figura 56 – Anexo 34 - Objeto relacionado a função de exclusão de cadastro [2/2] . . . .	62
Figura 57 – Anexo 35 - Objeto relacionado a função de Login do sistema [1/2] . . . . .	62
Figura 58 – Anexo 36 - Objeto relacionado a função de Login do sistema [2/2] . . . . .	63
Figura 59 – Anexo 37 - Objeto relacionado as funções do menu principal [1/3] . . . . .	63
Figura 60 – Anexo 38 - Objeto relacionado as funções do menu principal [2/3] . . . . .	63
Figura 61 – Anexo 39 - Objeto relacionado as funções do menu principal [3/3] . . . . .	64
Figura 62 – Anexo 40 - Objeto relacionado a função de adição de cadastro [1/2] . . . . .	64
Figura 63 – Anexo 41 - Objeto relacionado a função de adição de cadastro [2/2] . . . . .	64
Figura 64 – Anexo 42 - Objeto relacionado aos principais atributos do sistema [1/3] . . .	65
Figura 65 – Anexo 43 - Objeto relacionado aos principais atributos do sistema [2/3] . . .	65
Figura 66 – Anexo 44 - - Objeto relacionado aos principais atributos do sistema [3/3] . .	66
Figura 67 – Anexo 45 - Objeto relacionado as instruções de início do sistema . . . . .	66

## Sumário

<b>1</b>	<b>Introdução</b>	<b>10</b>
<b>1.1</b>	<b>Visão Geral</b>	<b>10</b>
<b>1.2</b>	<b>Principais Funcionalidades</b>	<b>10</b>
<b>2</b>	<b>Requisitos Funcionais</b>	<b>11</b>
<b>2.1</b>	<b>Login de Usuário</b>	<b>11</b>
<b>2.2</b>	<b>Pesquisar</b>	<b>11</b>
<b>2.3</b>	<b>Cadastro de Funcionários</b>	<b>11</b>
<b>2.4</b>	<b>Edição de Cadastro</b>	<b>11</b>
<b>2.5</b>	<b>Exclusão de Cadastro</b>	<b>11</b>
<b>2.6</b>	<b>Emissão de Relatórios</b>	<b>11</b>
<b>2.7</b>	<b>Emissão de Folhas de Pagamento</b>	<b>12</b>
<b>3</b>	<b>Requisitos Não-Funcionais</b>	<b>13</b>
<b>3.1</b>	<b>Segurança</b>	<b>13</b>
<b>3.2</b>	<b>Desempenho</b>	<b>13</b>
<b>3.3</b>	<b>Confiabilidade</b>	<b>13</b>
<b>3.4</b>	<b>Escalabilidade</b>	<b>13</b>
<b>3.5</b>	<b>Usabilidade</b>	<b>13</b>
<b>3.6</b>	<b>Acessibilidade</b>	<b>13</b>
<b>3.7</b>	<b>Conformidade Legal</b>	<b>13</b>
<b>3.8</b>	<b>Integração</b>	<b>14</b>
<b>3.9</b>	<b>Manutenção e Suporte</b>	<b>14</b>
<b>4</b>	<b>Termo de Consentimento para Tratamento de Dados</b>	<b>15</b>
<b>4.1</b>	<b>Dados Pessoais</b>	<b>15</b>
<b>4.2</b>	<b>Finalidades do Tratamento de Dados</b>	<b>15</b>
<b>4.3</b>	<b>Compartilhamento de Dados</b>	<b>16</b>
<b>4.4</b>	<b>Segurança dos Dados</b>	<b>16</b>
<b>4.5</b>	<b>Término do Tratamento dos Dados</b>	<b>16</b>
<b>4.6</b>	<b>Direitos do Titular</b>	<b>16</b>
<b>4.7</b>	<b>Direito de Revogação do Consentimento</b>	<b>17</b>
<b>5</b>	<b>Arquitetura e Design</b>	<b>18</b>
<b>5.1</b>	<b>Modelo de Desenvolvimento Utilizado</b>	<b>18</b>
<b>5.2</b>	<b>Benefícios de Desenvolvimento Incremental</b>	<b>18</b>
<b>6</b>	<b>Aplicação Desktop</b>	<b>20</b>

<b>6.1</b>	<b>Protótipos de Tela . . . . .</b>	<b>21</b>
<b>6.2</b>	<b>Banco de Dados . . . . .</b>	<b>24</b>
<b>6.3</b>	<b>Diagrama de Classes . . . . .</b>	<b>25</b>
<b>6.4</b>	<b>DataBase Microsoft SQL Server . . . . .</b>	<b>26</b>
<b>7</b>	<b>Aplicação Web . . . . .</b>	<b>27</b>
<b>7.1</b>	<b>Protótipos de Tela . . . . .</b>	<b>27</b>
<b>8</b>	<b>Aplicação Android . . . . .</b>	<b>29</b>
<b>9</b>	<b>Conclusão . . . . .</b>	<b>34</b>
	<b>Referências . . . . .</b>	<b>35</b>
	<b>ANEXOS</b>	<b>36</b>

## 1 Introdução

A gestão de folha de pagamentos é uma parte essencial das operações de recursos humanos de qualquer empresa, independentemente do seu tamanho. Garantir que os funcionários recebam pagamentos de forma precisa e pontual é crucial para manter a satisfação da equipe e cumprir as obrigações fiscais e trabalhistas. No entanto, o processamento manual da folha de pagamentos pode ser demorado, sujeito a erros e requer um esforço especial. É aqui que entra o Software de Automação de Folha de Pagamentos, uma solução poderosa que simplifica e agiliza esse processo fundamental. Esse tipo de software é capaz de automatizar uma série de tarefas relacionadas à folha de pagamento, desde o planejamento e os cálculos de encargos e deduções até a produção de relatórios financeiros e o cumprimento das obrigações legais.

### 1.1 Visão Geral

O projeto de Automação de Folha de Pagamentos é uma iniciativa estratégica que visa revolucionar a maneira como nossa empresa gerencia e processa funcionários de suas empresas. A automação da folha de pagamento é essencial para simplificar e aprimorar a gestão financeira e de recursos humanos, aumentando a eficiência operacional, a precisão e a conformidade com regulamentações trabalhistas e fiscais, garantindo que os pagamentos sejam pontuais e livres de erros.(SISTEMA. . . , )

### 1.2 Principais Funcionalidades

- Login de Usuário
- Pesquisar
- Cadastro de Funcionários
- Edição de Cadastro
- Exclusão de Cadastro
- Emissão de Relatórios
- Emissão de Folhas de Pagamento.

## 2 Requisitos Funcionais

### 2.1 Login de Usuário

A funcionalidade Login de Usuário fornece um acesso seguro ao sistema, garantindo a privacidade das informações. O usuário do sistema irá colocar seus dados de Login “Email e Senha” (“Usuário e Senha” na Aplicação Desktop), haverá acesso parcial para os Funcionários de RH e um acesso mais irrestrito a Gerência de RH e Outros Superiores.

### 2.2 Pesquisar

Essa funcionalidade possibilitará que o usuário pesquise por algum funcionário cadastrado em seu banco de dados, caso não possua o cadastro pesquisado o sistema irá informar uma mensagem de erro dizendo “Cadastro Não Encontrado”.

### 2.3 Cadastro de Funcionários

Essa funcionalidade permitirá ao usuário cadastrar um novo funcionário em seu banco de dados com seus principais dados pessoais e dados relacionados a empresa, tais como: *NOME, CPF, DATA DE NASCIMENTO, CARGO, NÚMERO DE REGISTRO, SALÁRIO*. Ao concluir esse processo o sistema irá mostrar uma mensagem “Cadastro Concluído com Sucesso!”.

### 2.4 Edição de Cadastro

Essa funcionalidade permitirá que o usuário faça uma alteração em qualquer dado informado DE algum cadastro armazenado em seu banco de dados, ao finalizar essa tarefa o sistema irá mostrar uma mensagem de “Edição Concluída com Sucesso!”.

### 2.5 Exclusão de Cadastro

Essa funcionalidade permitirá que o usuário exclua qualquer cadastro de funcionários incluído em seu banco de dados, o sistema irá localizar o cadastro e exibira uma mensagem “Tem certeza de que deseja excluir cadastro?” com duas opções SIM e NÃO, clicando na opção “sim” o sistema irá excluir o cadastro desejado e exibira a mensagem “Cadastro Excluído com Sucesso!”. Caso escolha a opção “não” o sistema voltara para a tabela de cadastro do funcionário.

### 2.6 Emissão de Relatórios

Essa funcionalidade permitirá que o usuário emita relatórios do funcionário cadastrado de sua escolha, com informações mensais de seus funcionários armazenados em seu banco de dados.

## **2.7 Emissão de Folhas de Pagamento**

Essa funcionalidade permitirá o usuário gerar a folha de pagamento de cada funcionário, onde será feito todos os cálculos trabalhistas de acordo com a leis e será fornecido a folha de pagamento que poderá ser impressa e entregue a cada funcionário para que tenha o seu próprio controle.

### **3 Requisitos Não-Funcionais**

#### **3.1 Segurança**

O *Software* deve ter medidas de seguranças robustas para proteger os dados pessoais dos funcionários, como criptografia, autenticação de dois fatores, acesso restrito aos dados e auditoria de registros.

#### **3.2 Desempenho**

O *Software* deve conseguir processar grandes volumes de dados de forma rápida e eficiente, garantindo que os cálculos de salários e benefícios sejam realizados dentro de prazos aceitáveis.

#### **3.3 Confiabilidade**

O *Software* deve ser confiável e estar disponível para uso sempre que necessário, minimizando o tempo de inatividade e evitando falhas críticas.

#### **3.4 Escalabilidade**

O *Software* deve ser capaz de lidar com o crescimento do número de funcionários e empresas atendidas, sem comprometer o desempenho ou a qualidade do serviço.

#### **3.5 Usabilidade**

O *Software* deve ser intuitivo e fácil de usar, com uma interface amigável que permita aos usuários navegarem e realizar tarefas de forma eficiente, mesmo sem treinamento extensivo.

#### **3.6 Acessibilidade**

O *Software* deve ser acessível a todos os usuários, independentemente de suas habilidades ou necessidades especiais, seguindo as diretrizes de acessibilidade.

#### **3.7 Conformidade Legal**

O *Software* deve estar em conformidade com as leis e regulamentações trabalhistas e fiscais, garantindo que os cálculos de salários, impostos e benefícios estejam corretos e atualizados.

### **3.8 Integração**

O *Software* deve ser capaz de se integrar com outros sistemas e aplicativos, como sistemas de recursos humanos, contabilidade e bancos, para facilitar a troca de informações e evitar a duplicação de dados.

### **3.9 Manutenção e Suporte**

O *Software* deve ser fácil de manter e atualizar, com suporte técnico disponível para resolver problemas e fornecer assistência aos usuários.

## 4 Termo de Consentimento para Tratamento de Dados

Este documento visa registrar a manifestação livre, pela qual o Titular concorda com o tratamento de seus dados pessoais para finalidade específica, em conformidade com a Lei nº 13.709 – Lei Geral de Proteção de Dados Pessoais (LGPD).

Ao manifestar sua aceitação para com o presente termo, o Titular consente que e concorda que a Controladora, tome decisões referentes ao tratamento de seus dados pessoais, bem como realize o tratamento de tais dados, envolvendo operações como as que se referem a coleta, produção, recepção, classificação, utilização, acesso, reprodução, transmissão, distribuição, processamento, arquivamento, armazenamento, eliminação, avaliação ou controle da informação, modificação, comunicação, transferência, difusão ou extração.(CONSELHO FEDERAL DE CONTABILIDADE, 07/10/2021)

### 4.1 Dados Pessoais

A Controladora fica autorizada a tomar decisões referentes ao tratamento e a realizar o tratamento dos seguintes dados pessoais do Titular:

- Nome completo;
- Data de nascimento;
- CPF;
- Cargo
- Número de registro do funcionário.

A controladora fica autorizada a utilizar os seguintes dados inseridos pelo Titular, com a intenção de obter a prestação dos serviços ofertados pela mesma, como, por exemplo, cálculo da folha de pagamento.

### 4.2 Finalidades do Tratamento de Dados

O tratamento dos dados pessoais listados neste termo tem a seguinte finalidade:

- Possibilita que a Controladora utilize tais dados para cadastrar funcionários no banco de dados.
- Possibilitar que a Controladora utilize tais dados na emissão de folha de pagamento de funcionários da empresa.
- Possibilitar que a Controladora utilize tais dados na elaboração de relatórios.

### 4.3 Compartilhamento de Dados

Os dados pessoais do titular não serão compartilhados com outros agentes de tratamento de dados, listadas neste termo, observados os princípios e as garantias estabelecidas pela Lei n.<sup>º</sup> 13.709.

### 4.4 Segurança dos Dados

A Controladora responsabiliza-se pela manutenção de medidas de segurança, técnicas e administrativas aptas a proteger os dados pessoais de acessos não autorizados e de situações acidentais ou ilícitas de destruição, perda, alteração, comunicação ou qualquer forma de tratamento inadequado, ou ilícito.

Em conformidade ao art. 48 da Lei nº 13.709, o Controlador comunicará ao Titular e à Autoridade Nacional de Proteção de Dados (ANPD) a ocorrência de incidente de segurança que possa acarretar risco ou dano relevante ao Titular.

### 4.5 Término do Tratamento dos Dados

A Controladora poderá manter e tratar os dados pessoais do Titular durante todo o período em que os mesmos forem pertinentes ao alcance das finalidades listadas neste termo. Dados pessoais anonimizados, sem possibilidade de associação ao indivíduo, poderão ser mantidos por período indefinido.

O Titular poderá solicitar via e-mail ou correspondência ao Controlador, a qualquer momento, que sejam eliminados os dados pessoais não anonimizados do Titular.

### 4.6 Direitos do Titular

O Titular tem direito a obter da Controladora, em relação aos dados por ela tratados, a qualquer momento e mediante requisição:

- I - Confirmação da existência de tratamento;
- II - Acesso aos dados;
- III - Correção de dados incompletos, inexatos ou desatualizados;
- IV - Anonimidade, bloqueio ou eliminação de dados desnecessários, excessivos ou tratados em desconformidade com o disposto na Lei n.<sup>º</sup> 13.709;
- V - Portabilidade dos dados a outro fornecedor de serviço ou produto, mediante requisição expressa e observados os segredos comercial e industrial, conforme a regulamentação do órgão controlador;

VI - Portabilidade dos dados a outro fornecedor de serviço ou produto, mediante requisição expressa, conforme a regulamentação da autoridade nacional, observados os segredos comercial e industrial;

VII - Eliminação dos dados pessoais tratados com o consentimento do titular, exceto nas hipóteses previstas no art. 16 da Lei nº 13.709;

VIII - Informação das entidades públicas e privadas com as quais o controlador realizou uso compartilhado de dados;

XI- Informação sobre a possibilidade de não fornecer consentimento e sobre as consequências da negativa;

X - Revogação do consentimento, nos termos do § 5º do art. 8º da Lei n.º 13.709.

#### **4.7 Direito de Revogação do Consentimento**

Este consentimento poderá ser revogado pelo Titular, a qualquer momento, mediante solicitação via e-mail ou correspondência ao Controlador.

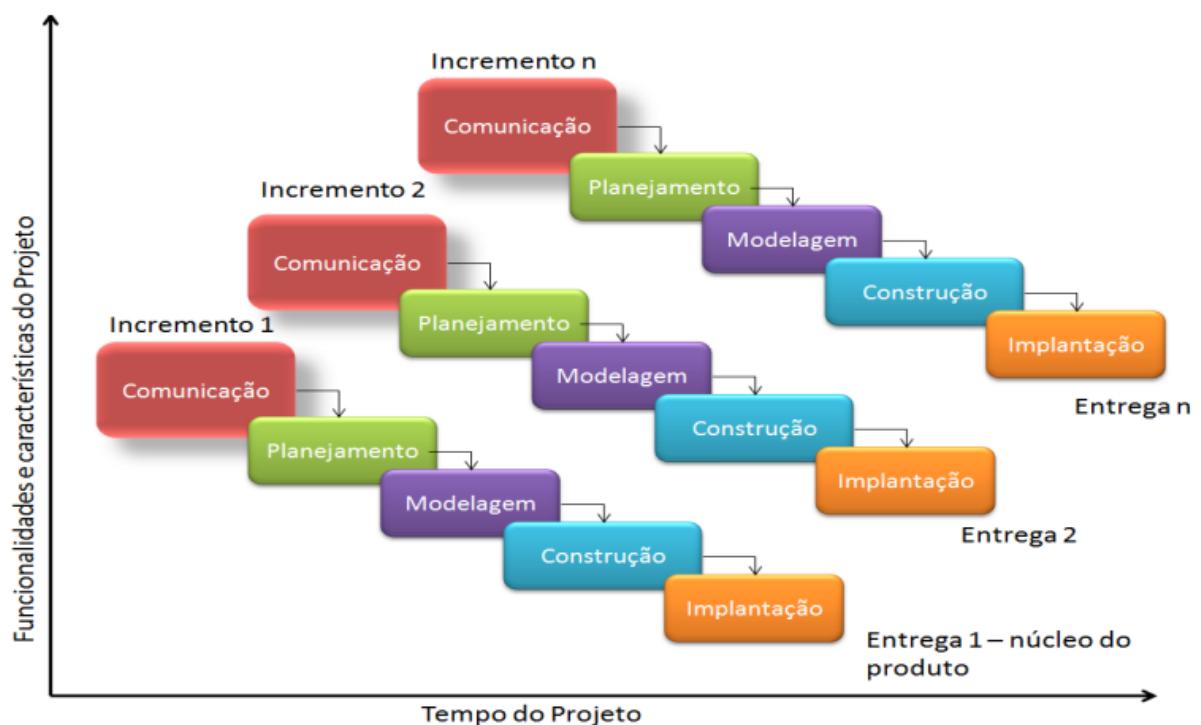
Eu concordo com os termos apresentados neste anexo para a finalidade de cadastrar e elaborar folha de pagamento dos funcionários.

## 5 Arquitetura e Design

### 5.1 Modelo de Desenvolvimento Utilizado

O Modelo Incremental surge como uma melhoria do Modelo em Cascata. Ao invés de especificar e desenvolver tudo de uma só vez, este modelo trabalha com incrementos, ou seja, pequenos pedaços de software entregues de cada vez. Este modelo combina elementos do Modelo em Cascata aplicados de maneira iterativa, ou seja, de forma que o progresso aconteça através de sucessivos refinamentos, melhorados a cada iteração. O primeiro incremento é frequentemente chamado de “núcleo do produto” e contém a implementação dos requisitos básicos para que o sistema possa funcionar e atender minimamente as necessidades do cliente. Cada aprimoramento é lançado como uma versão. Novas versões são criadas até que o sistema fique completo e adequado, para então, ser lançada a versão final.(Dias, 22/08/2019)

**Figura 1 – Exemplo de Modelo Incremental**



### 5.2 Benefícios de Desenvolvimento Incremental

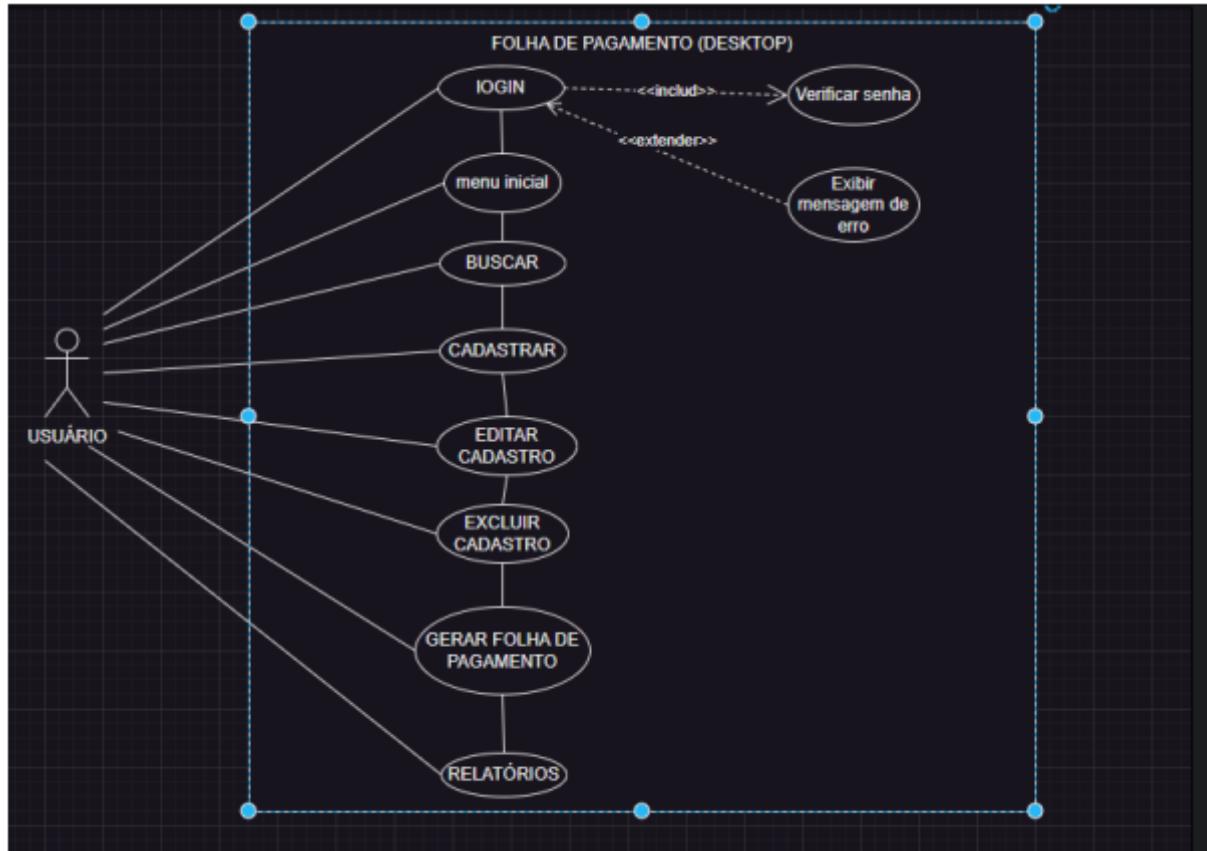
- Entrega Antecipada de Valor: O desenvolvimento incremental permite que as partes mais críticas ou valiosas do sistema sejam entregues rapidamente, possibilitando que os clientes obtenham benefícios reais mais cedo.
- Adaptação a Mudanças: A flexibilidade adicional ao desenvolvimento incremental permite que as equipes se adaptem às mudanças nos requisitos e nas prioridades, reduzindo o risco de projetos desatualizados.

- Maior Envolvimento dos Stakeholders: Com entregas regulares e *feedbacks*.
- Identificação Precoce de Problemas: Os problemas e as deficiências são identificados e abordados mais cedo, ocasionando a probabilidade de problemas significativos no final do projeto.
- Custos reduzidos: Ao liberar partes funcionais do software mais cedo, os custos de correção de erros e ajustes são menores que nas abordagens de desenvolvimento tradicionais.

## 6 Aplicação Desktop

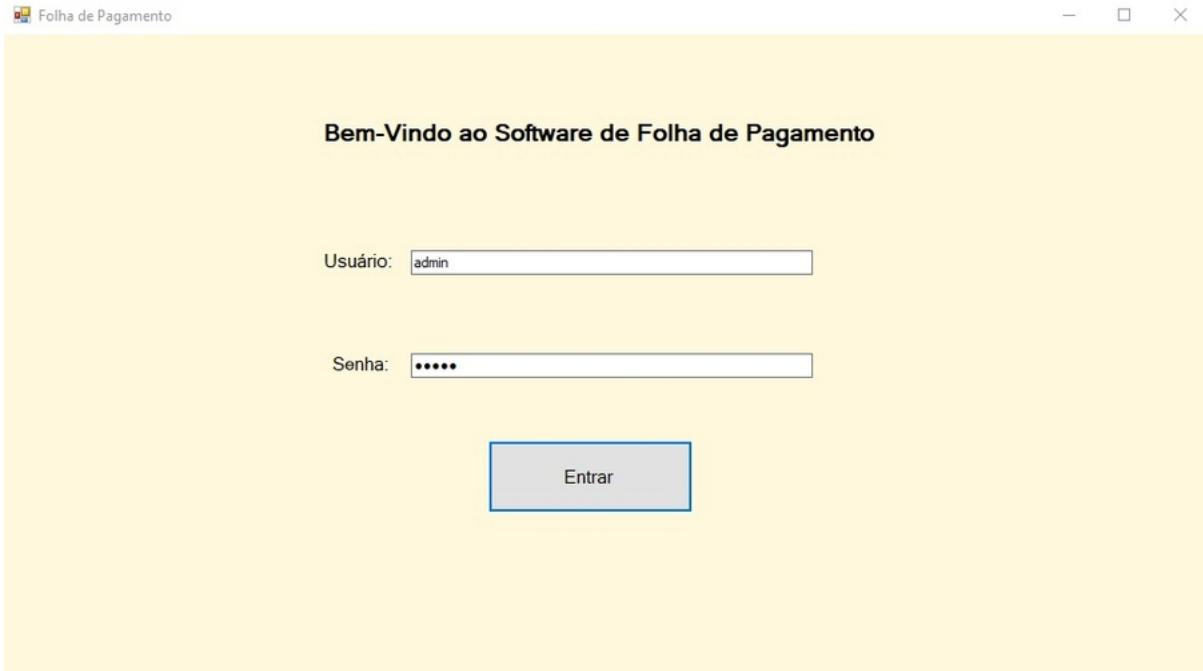
A Aplicação Desktop foi realizada utilizando a IDE Visual Studio 2022 utilizando a linguagem C# e SGBD SQL Server 2022, ambos proporcionados pela Microsoft.

Figura 2 – Caso de Uso demonstrando as funcionalidades da aplicação



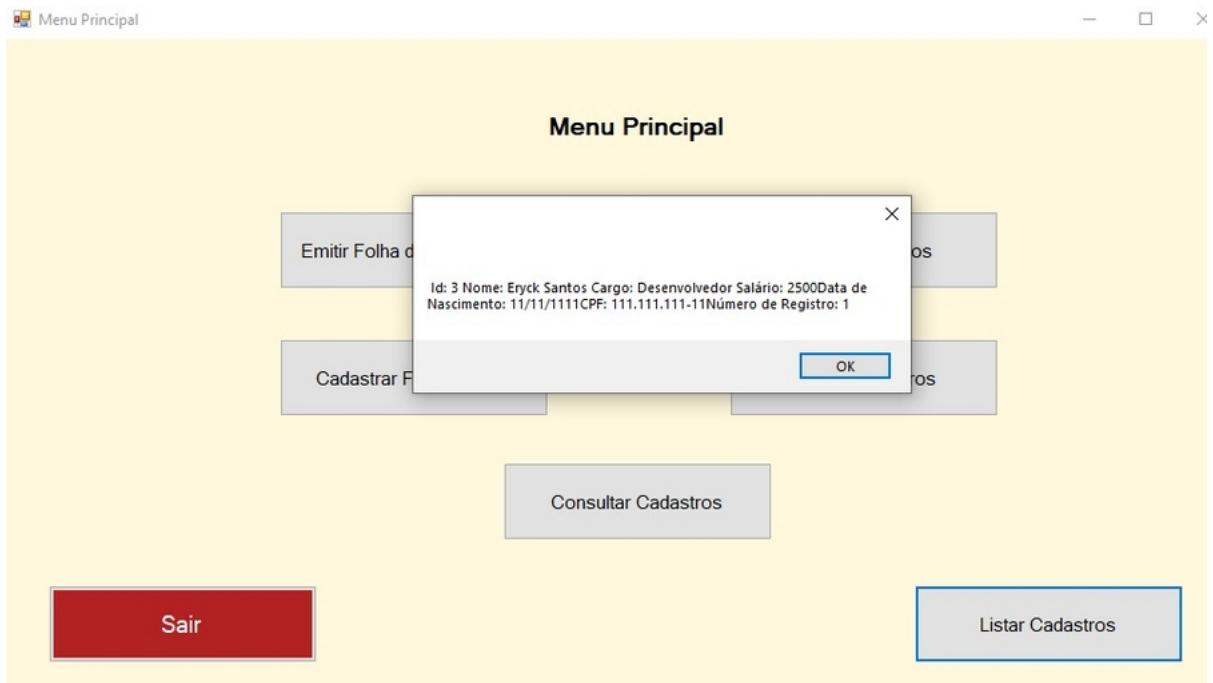
## 6.1 Protótipos de Tela

**Figura 3 – Tela de Login**

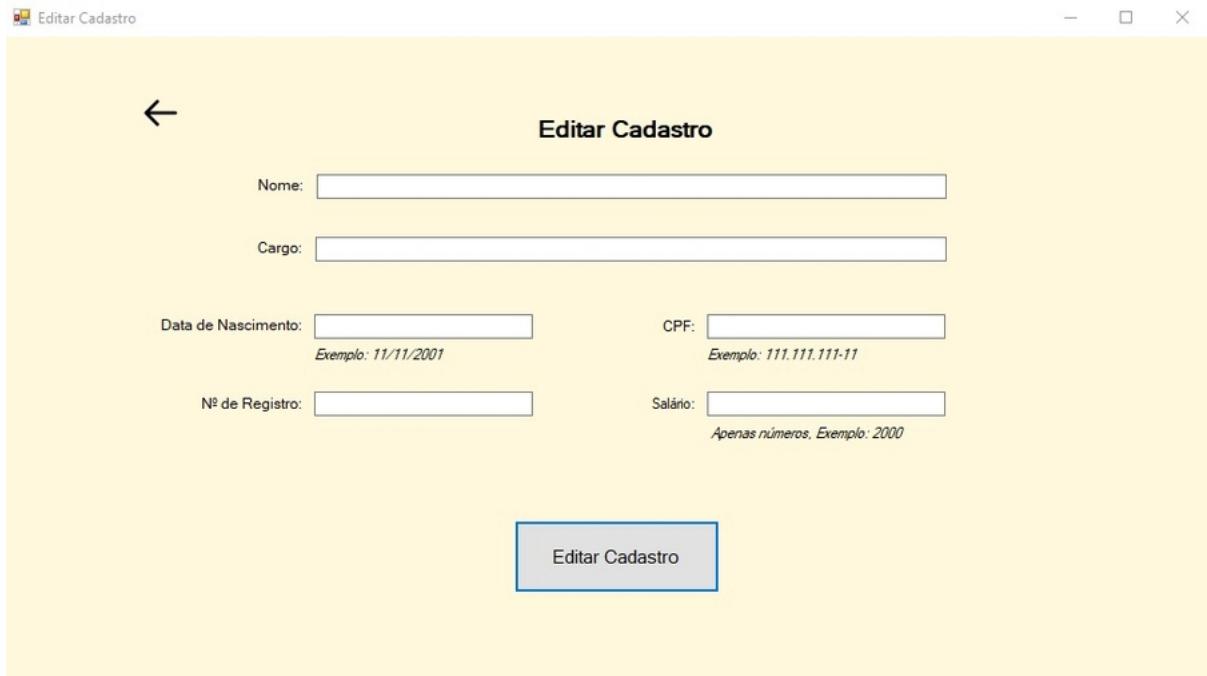


**Figura 4 – Menu Principal**



**Figura 5 – Relatório de Funcionários****Figura 6 – Cadastro de Funcionário**

The screenshot shows a Windows desktop application titled "Cadastrar Funcionário". It features a back arrow icon and the title "Cadastrar Funcionário". The form contains the following fields:  
Nome:   
Cargo:   
Data de Nascimento:  Exemplo: 01/01/2001  
CPF:  Exemplo: 111.111.111-11  
Nº de Registro:   
Salario:  Apenas números, Exemplo: 2000  
A "Cadastrar" button is located at the bottom right of the form area.

**Figura 7 – Edição de Cadastro****Figura 8 – Exclusão de Cadastro**

**Figura 9 – Consulta de Cadastro**

The screenshot shows a Windows application window titled "Consultar Cadastro". At the top left is a back arrow icon. The title bar has the window title and standard minimize, maximize, and close buttons. The main area contains several input fields and labels:

- Nome:**
- A small note below the name field says: "Digite apenas o Nome e clique no botão Consultar".
- Cargo:**
- Data de Nascimento:**
- CPF:**
- Nº de Registro:**
- Salário:**

In the center of the form is a large button labeled "Consultar".

**Figura 10 – Emissão de Folha de Pagamento**

The screenshot shows a Windows application window titled "Folha de Pagamento". At the top left is a back arrow icon. The title bar has the window title and standard minimize, maximize, and close buttons. The main area contains several input fields and labels:

- Nome:**  (containing "Eryck Santos")
- Cargo:**  (containing "Desenvolvedor")
- CPF:**  (containing "111.111.111-11")
- Data de Nascimento:**  (containing "11/11/1111")
- Nº de Registro:**  (containing "1")
- Salário Bruto:**  (containing "2500")
- Aliquota INSS:**  (containing "300")
- Vale-Transporte:**  (containing "150")
- Vale-Refeição:**  (containing "R\$ 30,00 / Dia")
- SALÁRIO LÍQUIDO:**  (containing "2050")

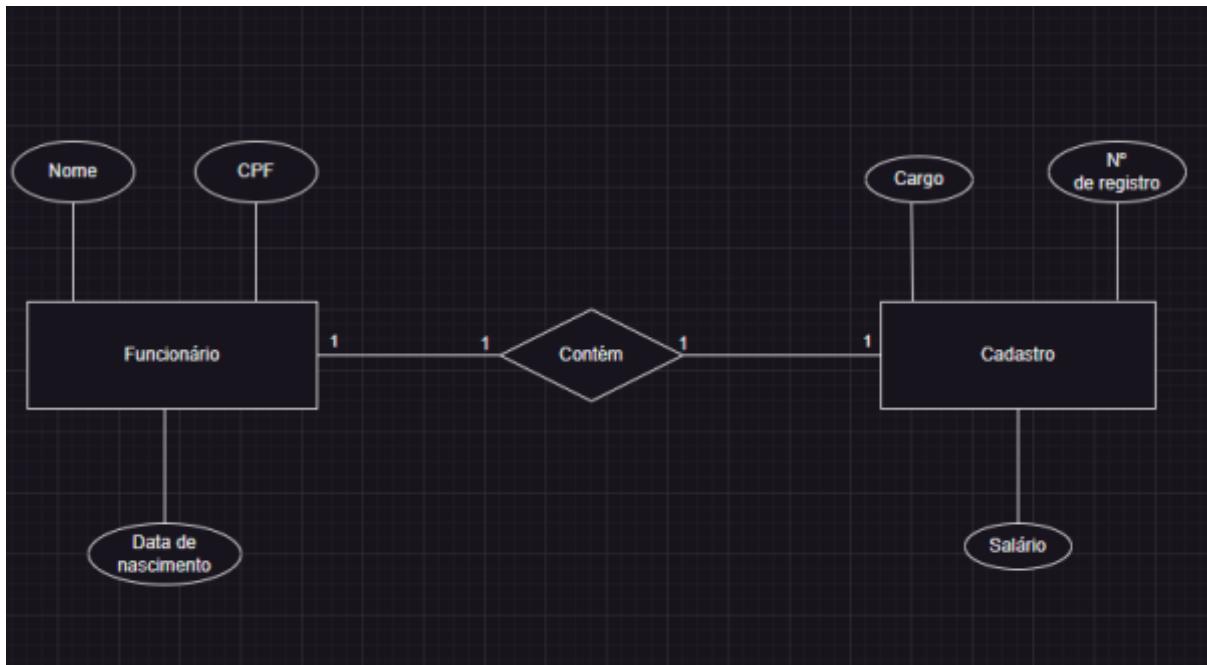
In the center of the form is a large button labeled "Emitir".

## 6.2 Banco de Dados

O Banco de Dados de um *Software* de Folha de Pagamento é um componente fundamental que armazena e gerencia todas as informações relacionadas aos funcionários de uma organização, suas obrigações, benefícios, impostos, deduções e outros aspectos financeiros essenciais. Ele desempenha um papel crucial na automatização e no gerenciamento eficiente do processo de

pagamento dos funcionários. Abaixo há exemplos de diagramas e tabelas de como o banco de dados desse software foi construído.

**Figura 11 – Diagrama DER do Banco de Dados**

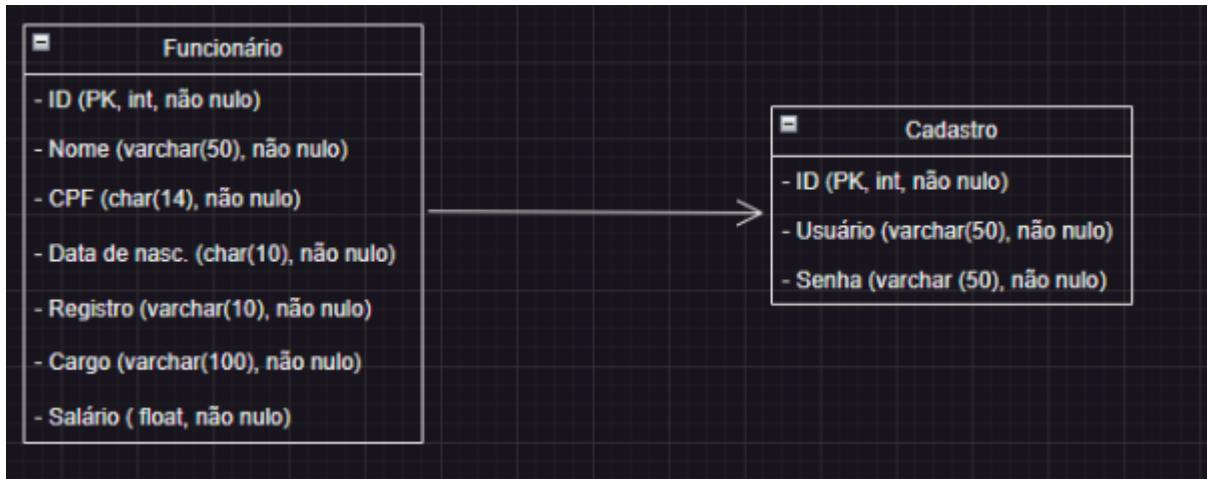


O Diagrama DER representado acima permite uma visualização mais clara sobre a estrutura do banco de dados, oferecem uma representação visual das tabelas, atributos e relacionamentos, tornando mais fácil para todos os envolvidos visualizar e discutir o modelo de dados.

### 6.3 Diagrama de Classes

A imagem abaixo representa o diagrama de classes do banco de dados onde foram extraídas as informações do diagrama DER e construída as tabelas com suas classes, atributos e relacionamentos.

Figura 12 – Diagrama de Classes



## 6.4 DataBase Microsoft SQL Server

Abaixo a representação das duas tabelas que compõem o Banco de Dados utilizado para este projeto, com os detalhes de cada coluna e seus respectivos comportamentos.

Figura 13 – Tabela com a representação das informações dos funcionários

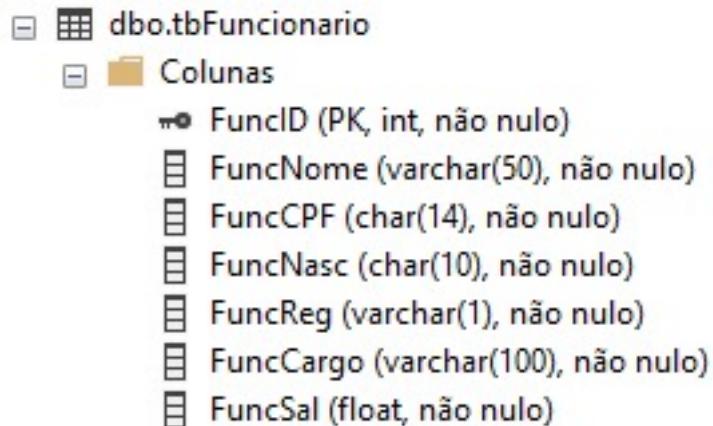
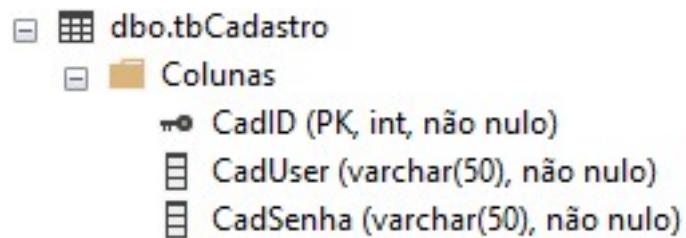


Figura 14 – Tabela representando as credenciais de Login do Sistema



## 7 Aplicação Web

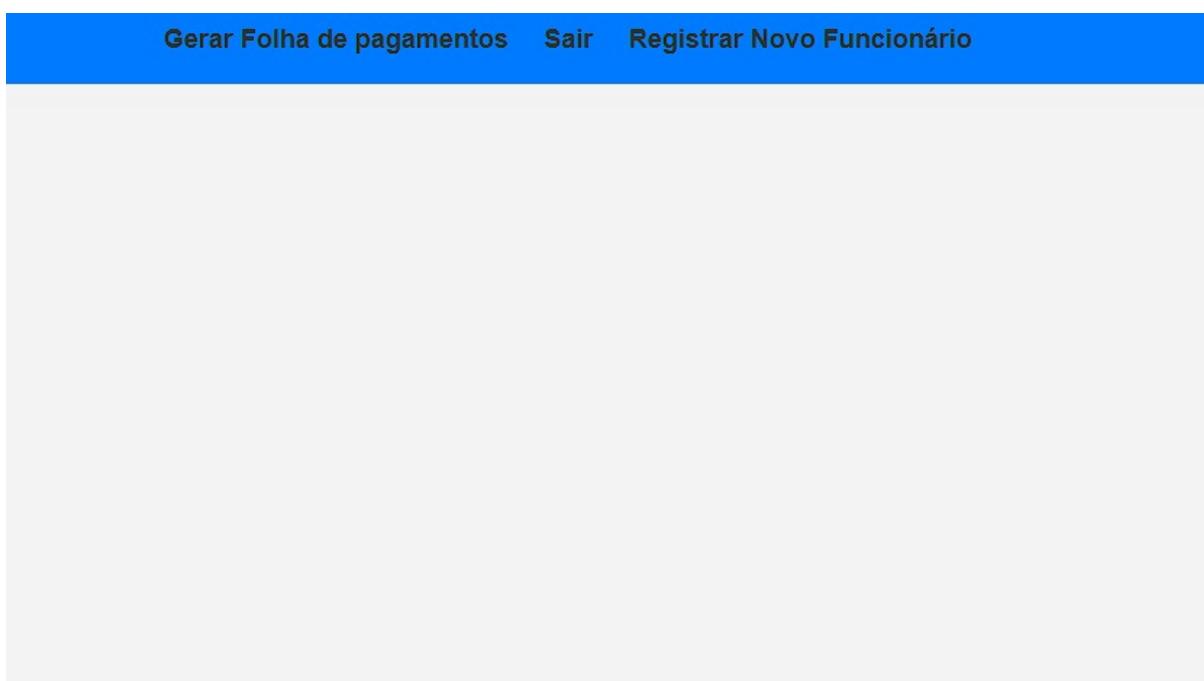
A Aplicação Web foi desenvolvida em HTML, CSS e JavaScript, com o uso do *Framework* Bootstrap para auxiliar na responsividade.

### 7.1 Protótipos de Tela

Figura 15 – Tela de Login da Aplicação Web



Figura 16 – Menu Principal da Aplicação Web



**Figura 17 – Cadastro de Funcionário na Aplicação Web**

Nome completo \_\_\_\_\_

Email \_\_\_\_\_

CPF \_\_\_\_\_

Cargo \_\_\_\_\_

Salário bruto (sem descontos) \_\_\_\_\_

Número de registro \_\_\_\_\_

**Registrar**

**Funcionários Cadastrados**

Nome Completo	Email	CPF	Cargo	Salário	Número de Registro	Ação
Funcionário não cadastrado						
Pesquisar funcionários		<b>Pesquisar</b>				

[Voltar para o Menu](#)**Figura 18 – Emissão de Folha de Pagamentos na Aplicação Web**

### Calculadora de Folha de Pagamento

Salário Bruto \_\_\_\_\_

Dias Trabalhados \_\_\_\_\_

Valor do VR \_\_\_\_\_

Valor do VT \_\_\_\_\_

Insalubridade:

Periculosidade:

**Calcular**

Salário Líquido: R\$ 0.00  
Desconto INSS: R\$ 0.00  
Desconto IRRF: R\$ 0.00  
Valor VR: R\$ 0.00  
Valor VT: R\$ 0.00  
Insalubridade: R\$ 0.00  
Periculosidade: R\$ 0.00

E-mail do Funcionário \_\_\_\_\_

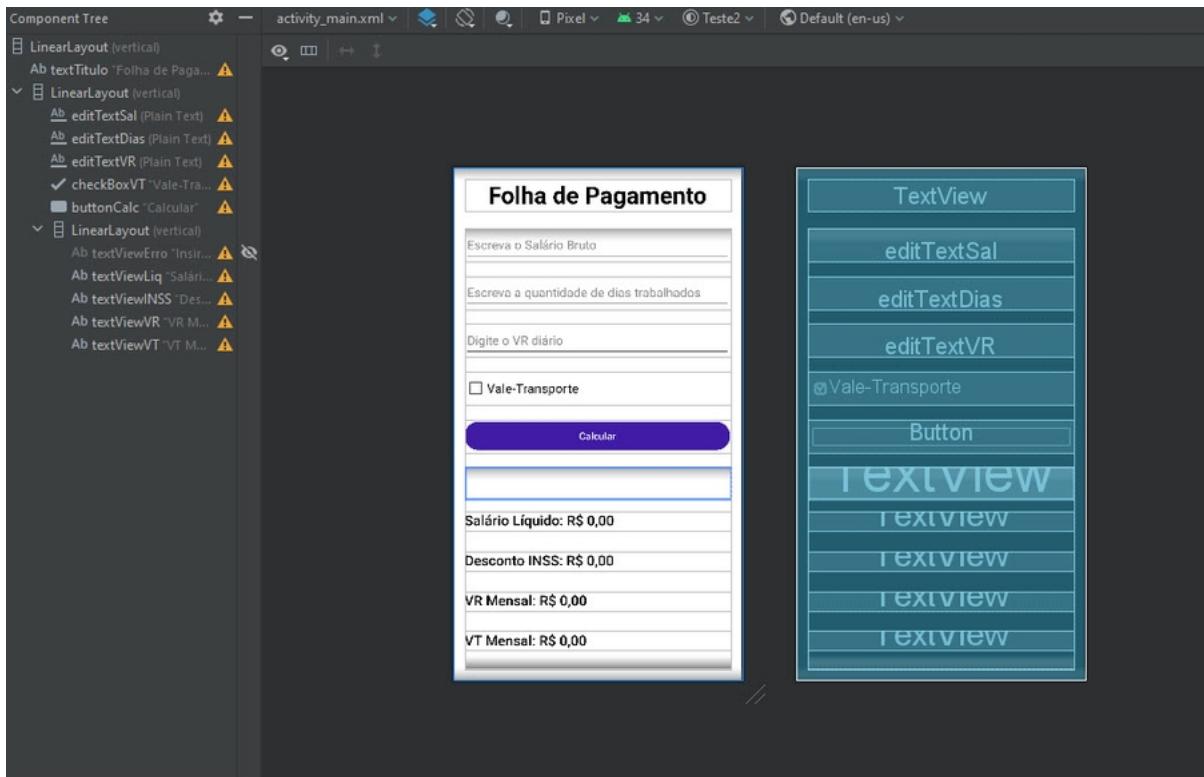
[Enviar por E-mail](#)

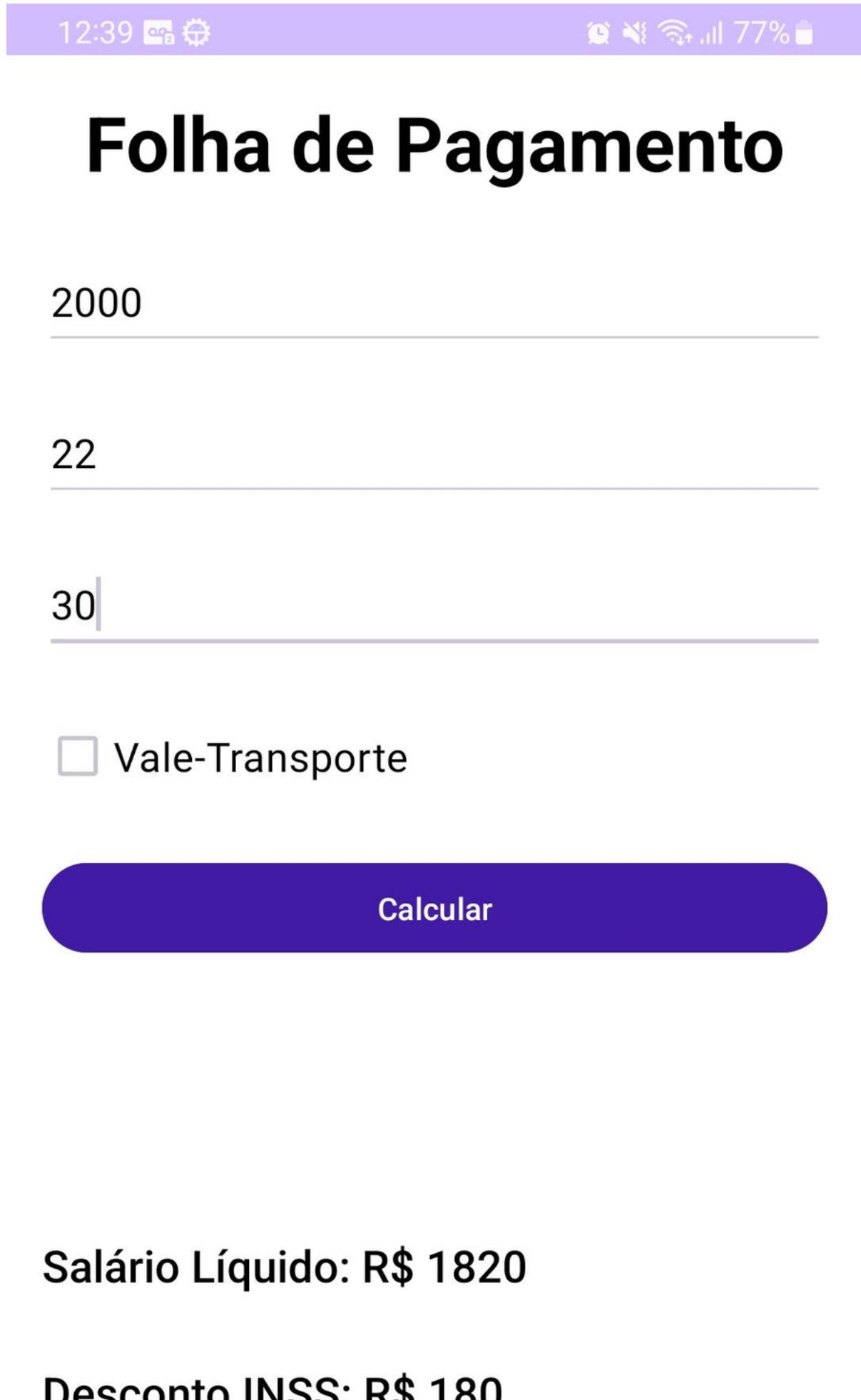
[Voltar para o menu](#)

## 8 Aplicação Android

A Aplicação Android do projeto foi realizado usando a IDE Android Studio, e desenvolvido na linguagem Java, é composta por uma calculadora básica que com 4 informações (salário bruto, quantidade de dias trabalhados, valor diário do Vale Refeição, e se há Vale Transporte ou não) calcula os descontos e ajusta os descontos referentes ao valor do salário bruto.

**Figura 19 – Layout do aplicativo**



**Figura 20 – Cálculo sem Vale-Transporte**



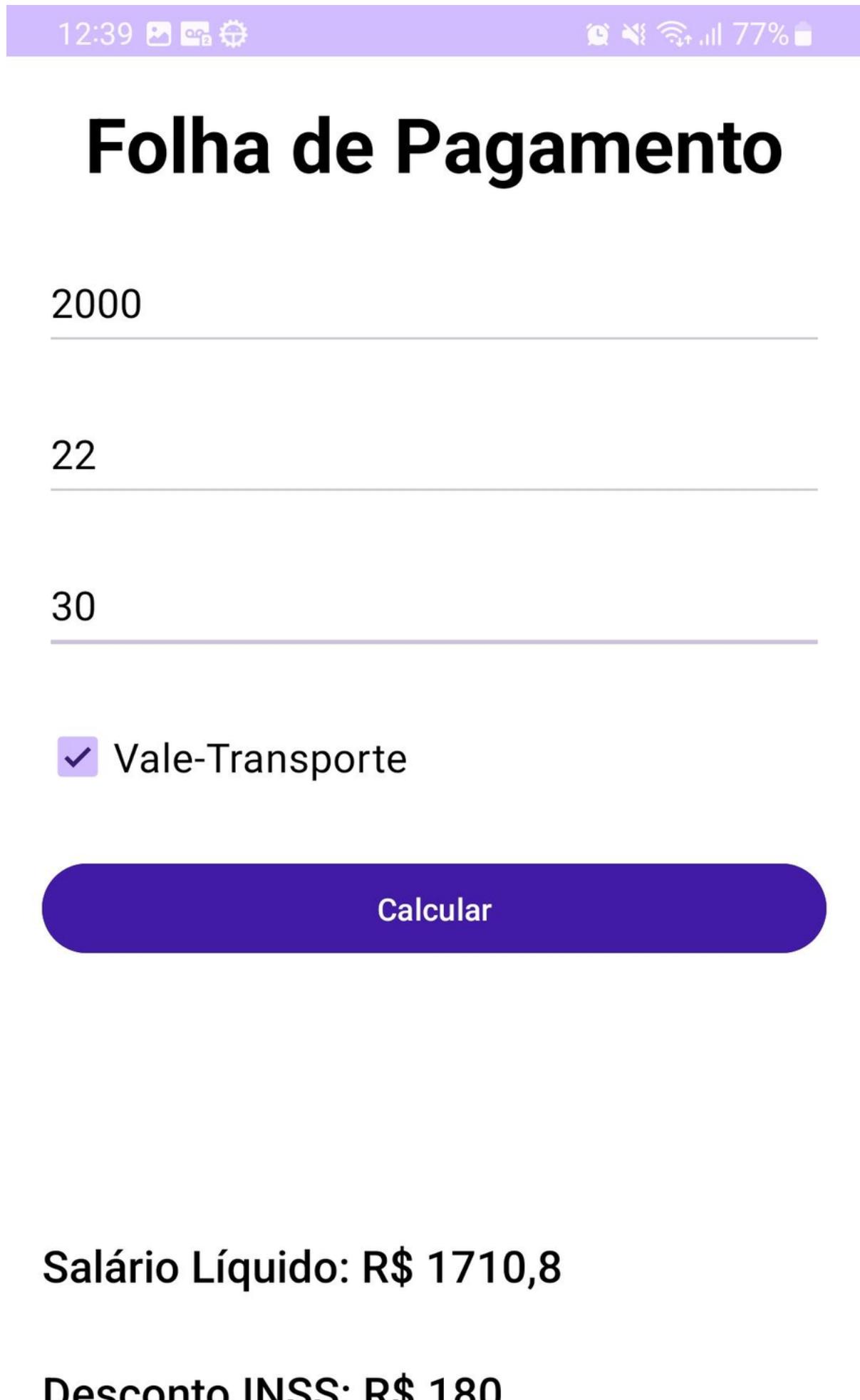
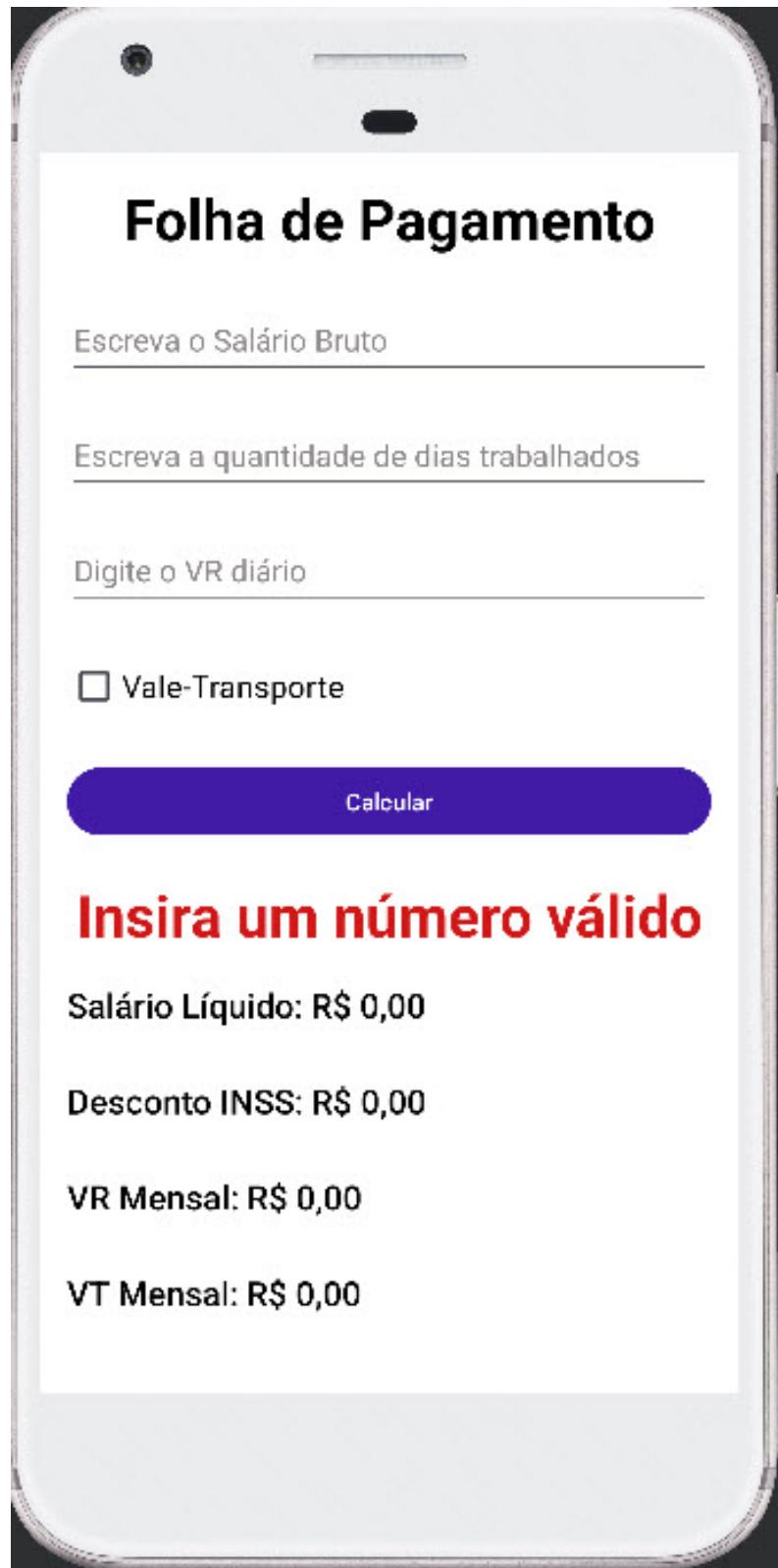
**Figura 21 – Cálculo com Vale-Transporte**

Figura 22 – Mensagem de erro



## 9 Conclusão

Por meio deste projeto foram apresentados argumentos, modelos e metodologias que serão usadas para a futura implementação do sistema de geração de folha de pagamentos. O resultado esperado é que o cliente entenda por este documento, o conceito e a especificação do sistema a ser implementado, onde a especificação do mesmo ajude-o a ter uma visão mais clara do projeto.

## Referências

- CONSELHO FEDERAL DE CONTABILIDADE. TERMO DE CONSENTIMENTO PARA TRATAMENTO DE DADOS PESSOAIS.** 07/10/2021. Disponível em: [https://cfc.org.br/wp-content/uploads/2021/10/Termo\\_Consentimento\\_final.pdf](https://cfc.org.br/wp-content/uploads/2021/10/Termo_Consentimento_final.pdf). Acesso em: 26/10/2023.
- DIAS, R. O Modelo Incremental.** 22/08/2019. Disponível em: <https://medium.com/contexto-delimitado/o-modelo-incremental-b41fc06cac04>. Acesso em: 26/10/2023.
- SISTEMA PARA FOLHA DE PAGAMENTO.** Disponível em: [https://exactus.com.br/solucoes/empresa-de-contabilidade/sistema-folha-de-pagamento/#av\\_section\\_2](https://exactus.com.br/solucoes/empresa-de-contabilidade/sistema-folha-de-pagamento/#av_section_2). Acesso em: 26/10/2023.

## **Anexos**

**Figura 23 – Anexo 1 - Código fonte da página de Login do App Web**

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="UTF-8">
5     <link rel="stylesheet" type="text/css" href="styles2.css">
6     <title>Bem-vindo à Folha de Pagamentos</title>
7 </head>
8 <body>
9     <div class="container">
10        <h1 class="title">Bem-vindo à Folha de Pagamentos</h1>
11        <form>
12            <!-- Email input -->
13            <div class="form-group">
14                <input type="email" id="emailInput" class="form-control" placeholder="Seu endereço de email">
15            </div>
16
17            <!-- Password input -->
18            <div class="form-group">
19                <input type="password" id="passwordInput" class="form-control" placeholder="Sua senha">
20            </div>
21
22            <!-- Botão de Login com função de redirecionamento -->
23            <button type="button" class="btn btn-primary" id="loginButton">Login</button>
24        </form>
25    </div>
26
27    <script>
28        // Defina o email e a senha permitidos
29        const allowedEmail = "kaykimolina@gmail.com";
30        const allowedPassword = "_____";
31
32        // Seleciona o botão de login pelo ID
33        const loginButton = document.getElementById("loginButton");
34
35        // Adiciona um ouvinte de clique ao botão de login
36        loginButton.addEventListener("click", function() {
37            // Obtém os valores dos campos de email e senha
38            const email = document.getElementById("emailInput").value;
39            const password = document.getElementById("passwordInput").value;
40
41            // Verifica se o email e a senha inseridos correspondem aos permitidos
42            if (email === allowedEmail && password === allowedPassword) {
43                // Redireciona para a página de sucesso (substitua a URL pela sua página de sucesso)
44                window.location.href = "index4.html";
45            } else {
46                // Exibe uma mensagem de erro se as credenciais estiverem incorretas
47                alert("Credenciais incorretas. Tente novamente.");
48            }
49        });
50    </script>
51 </body>
52 </html>
53
```

**Figura 24 – Anexo 2 - Código fonte do Menu Principal do App Web**

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="UTF-8">
5     <link rel="stylesheet" type="text/css" href="styles4.css">
6     <title>Menu</title>
7 </head>
8 <body>
9     <div class="menu">
10         <ul>
11             <li><a href="#" id="gerarFolha">Gerar Folha de pagamentos</a></li>
12             <li><a href="#" id="sair">Sair</a></li>
13             <li><a href="#" id="registrarNovoFuncionario">Registrar Novo Funcionário</a></li>
14         </ul>
15     </div>
16
17     <script>
18         // Adicionar event listeners para as opções do menu
19         document.getElementById("gerarFolha").addEventListener("click", function() {
20             // Redirecionar para a página de geração de folha de pagamento
21             window.location.href = "index.html";
22         });
23
24         document.getElementById("sair").addEventListener("click", function() {
25             // Realizar a ação de sair (por exemplo, encerrar a sessão)
26             alert("Saindo da sua conta..."); // Você pode adicionar sua lógica de saída aqui
27             window.location.href = "index2.html";
28         });
29         // Adicionar event listeners para as opções do menu
30         document.getElementById("registrarNovoFuncionario").addEventListener("click", function() {
31             window.location.href = "index5.html";
32         });
33     </script>
34 </body>
35 </html>
```

**Figura 25 – Anexo 3 - Código fonte da página de cadastro do App Web [1/4]**

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <link rel="stylesheet" type="text/css" href="styles5.css">
6   </head>
7   <body>
8     <form>
9
10       <div class="form-outline mb-4">
11         <input type="text" id="fullName" class="form-control" />
12         <label class="form-label" for="fullName">Nome completo</label>
13       </div>
14
15
16       <div class="form-outline mb-4">
17         <input type="text" id="email" class="form-control" />
18         <label class="form-label" for="email">Email</label>
19       </div>
20
21
22       <div class="form-outline mb-4">
23         <input type="text" id="cpf" class="form-control" />
24         <label class="form-label" for="cpf">CPF</label>
25       </div>
26
27       <div class="form-outline mb-4">
28         <input type="text" id="cargo" class="form-control" />
29         <label class="form-label" for="cargo">Cargo</label>
30       </div>
31
32
33       <div class="form-outline mb-4">
34         <input type="text" id="salario" class="form-control" />
35         <label class="form-label" for="salario">Salário bruto (sem descontos)</label>
36       </div>
37
38
39       <div class="form-outline mb-4">
40         <input type="text" id="numeroRegistro" class="form-control" />
41         <label class="form-label" for="numeroRegistro">Número de registro</label>
42       </div>
43
44
45         <button type="button" class="btn btn-primary btn-block mb-4" id="registerButton">Registrar</button>
46     </form>
47
```

**Figura 26 – Anexo 4 - Código fonte da página de cadastro do App Web [2/4]**

```
47 <h2>Funcionários Cadastrados</h2>
48 <table>
49   <tr>
50     <th>Nome Completo</th>
51     <th>Email</th>
52     <th>CPF</th>
53     <th>Cargo</th>
54     <th>Salário</th>
55     <th>Número de Registro</th>
56     <th>Ação</th>
57   </tr>
58   <tbody id="employeeList">
59
60   </tbody>
61 </table>
62
63
64 <div class="search-container">
65   <input type="text" id="searchEmployee" class="form-control" placeholder="Pesquisar func:>
66   <button type="button" class="btn btn-primary" id="searchButton">Pesquisar</button>
67 </div>
68
69 <script>
70   const registerButton = document.getElementById("registerButton");
71   const searchButton = document.getElementById("searchButton");
72   const searchEmployeeInput = document.getElementById("searchEmployee");
73   const employeeList = document.getElementById("employeeList");
74   let employees = getEmployeesFromLocalStorage() || [];
75
76   function getEmployeesFromLocalStorage() {
77     const storedEmployees = localStorage.getItem('employees');
78     if (storedEmployees) {
79       return JSON.parse(storedEmployees);
80     }
81     return [];
82   }
83
84   function saveEmployeesToLocalStorage() {
85     localStorage.setItem('employees', JSON.stringify(employees));
86   }
87
88   function updateEmployeeList() {
89     saveEmployeesToLocalStorage();
90     displayEmployees(employees);
91   }
92
93   registerButton.addEventListener("click", function() {
94     const fullName = document.getElementById("fullName").value;
95     const email = document.getElementById("email").value;
96     const cpf = document.getElementById("cpf").value;
97     const cargo = document.getElementById("cargo").value;
98     const salario = document.getElementById("salario").value;
99     const numeroRegistro = document.getElementById("numeroRegistro").value;
100
```

Figura 27 – Anexo 5 - Código fonte da página de cadastro do App Web [3/4]

```
102     const employee = {
103         fullName: fullName,
104         email: email,
105         cpf: cpf,
106         cargo: cargo,
107         salario: salario,
108         numeroRegistro: numeroRegistro
109     };
110
111     employees.push(employee);
112
113     document.getElementById("fullName").value = "";
114     document.getElementById("email").value = "";
115     document.getElementById("cpf").value = "";
116     document.getElementById("cargo").value = "";
117     document.getElementById("salario").value = "";
118     document.getElementById("numeroRegistro").value = "";
119
120     updateEmployeeList();
121 });
122
123 searchButton.addEventListener("click", function() {
124     const searchTerm = searchEmployeeInput.value.toLowerCase();
125     const filteredEmployees = employees.filter(function(employee) {
126         return (
127             employee.fullName.toLowerCase().includes(searchTerm) ||
128             employee.email.toLowerCase().includes(searchTerm) ||
129             employee.cpf.includes(searchTerm) ||
130             employee.cargo.toLowerCase().includes(searchTerm) ||
131             employee.salario.includes(searchTerm) ||
132             employee.numeroRegistro.includes(searchTerm)
133         );
134     });
135
136     displayEmployees(filteredEmployees);
137 });
138
139 function displayEmployees(employeesToDisplay) {
140     employeeList.innerHTML = "";
141
142     if (employeesToDisplay.length === 0) {
143         const row = document.createElement("tr");
144         row.innerHTML = `<td colspan="7">Funcionário não cadastrado</td>`;
145         employeeList.appendChild(row);
146     } else {
147         employeesToDisplay.forEach(function(employee, index) {
148             const row = document.createElement("tr");
149             row.innerHTML =
150                 `<td>${employee.fullName}</td>
151                 <td>${employee.email}</td>
152                 <td>${employee.cpf}</td>
153                 <td>${employee.cargo}</td>
154                 <td>${employee.salario}</td>
155                 <td>${employee.numeroRegistro}</td>
```

**Figura 28 – Anexo 6 - Código fonte da página de cadastro do App Web [4/4]**

```
156     <td>
157         <button onclick="editEmployee(${index})">Editar</button>
158         <button onclick="deleteEmployee(${index})">Excluir</button>
159     </td>
160     `;
161     employeeList.appendChild(row);
162   );
163 }
164
165 function editEmployee(index) {
166   const employee = employees[index];
167   document.getElementById("fullName").value = employee.fullName;
168   document.getElementById("email").value = employee.email;
169   document.getElementById("cpf").value = employee.cpf;
170   document.getElementById("cargo").value = employee.cargo;
171   document.getElementById("salario").value = employee.salario;
172   document.getElementById("numeroRegistro").value = employee.numeroRegistro;
173
174   employees.splice(index, 1);
175   updateEmployeeList();
176 }
177
178 function deleteEmployee(index) {
179   employees.splice(index, 1);
180   updateEmployeeList();
181 }
182
183 displayEmployees(employees);
184 </script>
185
186 <a href="index4.html">Voltar para o Menu</a>
187 </body>
188 </html>
189
190
```

**Figura 29 – Anexo 7 - Código fonte da página de geração de folha de pagamentos [1/2]**

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="UTF-8">
5     <title>Calculadora de Folha de Pagamento</title>
6     <link rel="stylesheet" type="text/css" href="styles.css">
7 </head>
8 <body>
9     <h1>Calculadora de Folha de Pagamento</h1>
10    <div id="calculator">
11        <input type="number" id="salarioBruto" placeholder="Salário Bruto">
12        <input type="number" id="diasTrabalhados" placeholder="Dias Trabalhados">
13        <input type="number" id="valorVR" placeholder="Valor do VR">
14        <input type="number" id="valorVT" placeholder="Valor do VT">
15        <label for="insalubridade">Insalubridade:</label>
16        <input type="checkbox" id="insalubridade">
17        <label for="periculosidade">Periculosidade:</label>
18        <input type="checkbox" id="periculosidade">
19        <button onclick="calcularFolha()">Calcular</button>
20    </div>
21    <div id="result">
22        <div id="resultText">
23            <p>Salário Líquido: <span id="salarioLiquido">R$ 0.00</span></p>
24            <p>Desconto INSS: <span id="descontoINSS">R$ 0.00</span></p>
25            <p>Desconto IRRF: <span id="descontoIRRF">R$ 0.00</span></p>
26            <p>Valor VR: <span id="valorVRCalculado">R$ 0.00</span></p>
27            <p>Valor VT: <span id="valorVTCalculado">R$ 0.00</span></p>
28            <p>Insalubridade: <span id="insalubridadeCalculada">R$ 0.00</span></p>
29            <p>Periculosidade: <span id="periculosidadeCalculada">R$ 0.00</span></p>
30        </div>
31    </div>
32
33    <!-- Adicione este campo no formulário de cálculo da folha de pagamento -->
34    <input type="email" id="emailFuncionario" placeholder="E-mail do Funcionário">
35
36    <!-- Adicione o botão de envio de e-mail -->
37    <button onclick="enviarFolhaPorEmail()">Enviar por E-mail</button>
38
39    <!-- Botão "Voltar" -->
40    <a href="#">index4.html>Voltar para o menu</a>
41
42    <script>
43        function calcularFolha() {
44            const salarioBruto = parseFloat(document.getElementById("salarioBruto").value);
45            const descontoINSS = calcularINSS(salarioBruto);
46            const descontoIRRF = calcularIRRF(salarioBruto - descontoINSS);
47
48            const diasTrabalhados = parseInt(document.getElementById("diasTrabalhados").value);
49            const valorVR = parseFloat(document.getElementById("valorVR").value);
50            const valorVT = parseFloat(document.getElementById("valorVT").value);
51            const insalubridade = document.getElementById("insalubridade").checked ? 1.2 : 1;
52            const periculosidade = document.getElementById("periculosidade").checked ? 1.3 : 1;
53        }
54    </script>
```

**Figura 30 – Anexo 8 - Código fonte da página de geração de folha de pagamentos [2/2]**

```
53
54     const salarioBase = salarioBruto - descontoINSS - descontoIRRF;
55     const salarioComInsalubridade = salarioBase * insalubridade;
56     const salarioComPericulosidade = salarioBase * periculosidade;
57     const valorVRCalculado = valorVR * diasTrabalhados;
58     const valorVTCalculado = valorVT * diasTrabalhados;
59
60     const salarioLiquido = salarioComPericulosidade + salarioComInsalubridade - valorVRCalculado - valorVTCalculado;
61
62     document.getElementById("salarioLiquido").textContent = `R$ ${salarioLiquido.toFixed(2)}`;
63     document.getElementById("descontoINSS").textContent = `R$ ${descontoINSS.toFixed(2)}`;
64     document.getElementById("descontoIRRF").textContent = `R$ ${descontoIRRF.toFixed(2)}`;
65     document.getElementById("insalubridadeCalculada").textContent = `R$ ${((salarioBase * (insalubridade - 1))).toFixed(2)}`;
66     document.getElementById("periculosidadeCalculada").textContent = `R$ ${((salarioBase * (periculosidade - 1))).toFixed(2)}`;
67     document.getElementById("valorVRCalculado").textContent = `R$ ${valorVRCalculado.toFixed(2)}`;
68     document.getElementById("valorVTCalculado").textContent = `R$ ${valorVTCalculado.toFixed(2)}`;
69   }
70
71   function calcularINSS(salarioBruto) {
72     // Substitua essas taxas de exemplo pelas taxas reais do INSS do seu país
73     const taxaINSS = 0.1; // 10%
74     return salarioBruto * taxaINSS;
75   }
76
77   function calcularIRRF(salarioBase) {
78     // Substitua essas taxas de exemplo pelas taxas reais do IRRF do seu país
79     const taxaIRRF = 0.15; // 15%
80     return salarioBase * taxaIRRF;
81   }
82
83   function enviarFolhaPorEmail() {
84     // Recupere o e-mail do funcionário e as informações da folha de pagamento
85     const emailFuncionario = document.getElementById("emailFuncionario").value;
86     const salarioLiquido = document.getElementById("salarioLiquido").textContent;
87     const descontoINSS = document.getElementById("descontoINSS").textContent;
88     const descontoIRRF = document.getElementById("descontoIRRF").textContent;
89     const valorVR = document.getElementById("valorVRCalculado").textContent;
90     const valorVT = document.getElementById("valorVTCalculado").textContent;
91     const insalubridade = document.getElementById("insalubridadeCalculada").textContent;
92     const periculosidade = document.getElementById("periculosidadeCalculada").textContent;
93
94     // Construa o corpo do e-mail com as informações da folha de pagamento
95     const corpoEmail = `
96       Aqui estão as informações da folha de pagamento:
97       Salário Líquido: ${salarioLiquido}
98       Desconto INSS: ${descontoINSS}
99       Desconto IRRF: ${descontoIRRF}
100      Valor VR Calculado: ${valorVR}
101      Valor VT Calculado: ${valorVT}
102      Insalubridade Calculada: ${insalubridade}
103      Periculosidade Calculada: ${periculosidade}
104    `;
105
106    // Crie um link de e-mail com o corpo do e-mail preenchido
107    const emailLink = `mailto:${emailFuncionario}?subject=Folha de Pagamento&body=${encodeURIComponent(corpoEmail)}`;
108
109    // Abra o cliente de e-mail padrão com o link de e-mail preenchido
110    window.location.href = emailLink;
111  }
112  </script>
113 </body>
114 </html>
```

**Figura 31 – Anexo 9 - Representação em Código XML do Layout do App Android [1/6]**

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent"
6      android:background="#ffffffff"
7      android:backgroundTint="#ffffffff"
8      android:orientation="vertical"
9      android:padding="16dp">
10
11     <TextView
12         android:id="@+id/textTitulo"
13         android:layout_width="match_parent"
14         android:layout_height="wrap_content"
15         android:layout_marginBottom="25dp"
16         android:text="Folha de Pagamento"
17         android:textAlignment="center"
18         android:textAppearance="@style/TextAppearance.AppCompat.Display1"
19         android:textColor="@color/black"
20         android:textStyle="bold" />
21
22     <LinearLayout
23         android:layout_width="match_parent"
24         android:layout_height="match_parent"
25         android:background="@color/white"
26         android:orientation="vertical">
```

**Figura 32 – Anexo 10 - Representação em Código XML do Layout do App Android [2/6]**

```
28     <EditText  
29         android:id="@+id/editTextSal"  
30         android:layout_width="match_parent"  
31         android:layout_height="wrap_content"  
32         android:layout_marginBottom="20dp"  
33         android:ems="10"  
34         android:hint="Escreva o Salário Bruto"  
35         android:inputType="text"  
36         android:minHeight="48dp"  
37         android:textColor="#000000"  
38         android:textColorHint="#8A8787" />  
39  
40  
41     <EditText  
42         android:id="@+id/editTextDias"  
43         android:layout_width="match_parent"  
44         android:layout_height="wrap_content"  
45         android:layout_marginBottom="20dp"  
46         android:ems="10"  
47         android:hint="Escreva a quantidade de dias trabalhados"  
48         android:inputType="text"  
49         android:minHeight="48dp"  
50         android:textColor="#000000"  
51         android:textColorHint="#8A8787" />  
52
```

Figura 33 – Anexo 11 - Representação em Código XML do Layout do App Android [3/6]

```
52
53     <EditText
54         android:id="@+id/editTextVR"
55         android:layout_width="match_parent"
56         android:layout_height="wrap_content"
57         android:layout_marginBottom="20dp"
58         android:ems="10"
59         android:hint="Digite o VR diário"
60         android:inputType="text"
61         android:minHeight="48dp"
62         android:textColor="#000000"
63         android:textColorHint="#8A8787" />
64
65     <CheckBox
66         android:id="@+id/checkBoxVT"
67         android:layout_width="match_parent"
68         android:layout_height="wrap_content"
69         android:layout_marginBottom="20dp"
70         android:text="Vale-Transporte"
71         android:textColor="#000000"
72         android:textSize="18dp" />
73
74     <Button
75         android:id="@+id/buttonCalc"
76         android:layout_width="match_parent"
77         android:layout_height="wrap_content"
78         android:layout_marginBottom="20dp"
79         android:backgroundTint="#411BA6"
80         android:onClick="calcularFolha"
81         android:text="Calcular"
82         android:textColor="#fffff" />
```

**Figura 34 – Anexo 12 - Representação em Código XML do Layout do App Android [4/6]**

```
84      <LinearLayout  
85          android:layout_width="match_parent"  
86          android:layout_height="match_parent"  
87          android:orientation="vertical">  
88  
89  
90          <TextView  
91              android:id="@+id/textViewErro"  
92              android:layout_width="match_parent"  
93              android:layout_height="wrap_content"  
94              android:text="Insira um número válido"  
95              android:textAlignment="center"  
96              android:textColor="#D11818"  
97              android:textSize="34sp"  
98              android:textStyle="bold"  
99              android:visibility="invisible" />  
100  
101         <TextView  
102             android:id="@+id/textViewLiq"  
103             android:layout_width="match_parent"  
104             android:layout_height="wrap_content"  
105             android:layout_marginTop="17dp"  
106             android:layout_marginBottom="20dp"  
107             android:text="Salário Líquido: R$ 0,00"  
108             android:textAppearance="@style/TextAppearance.AppCompat.Body2"  
109             android:textColor="#000000"  
110             android:textSize="20sp" />
```

**Figura 35 – Anexo 13 - Representação em Código XML do Layout do App Android [5/6]**

```
112     <TextView  
113         android:id="@+id/textViewINSS"  
114         android:layout_width="match_parent"  
115         android:layout_height="wrap_content"  
116         android:layout_marginTop="10dp"  
117         android:layout_marginBottom="20dp"  
118         android:text="Desconto INSS: R$ 0,00"  
119         android:textAppearance="@style/TextAppearance.AppCompat.Body2"  
120         android:textColor="#000000"  
121         android:textSize="20sp" />  
122  
123     <TextView  
124         android:id="@+id/textViewVR"  
125         android:layout_width="match_parent"  
126         android:layout_height="wrap_content"  
127         android:layout_marginTop="10dp"  
128         android:layout_marginBottom="20dp"  
129         android:text="VR Mensal: R$ 0,00"  
130         android:textAppearance="@style/TextAppearance.AppCompat.Body2"  
131         android:textColor="#000000"  
132         android:textSize="20sp" />  
133  
134     <TextView  
135         android:id="@+id/textViewVT"  
136         android:layout_width="match_parent"  
137         android:layout_height="wrap_content"  
138         android:layout_marginTop="10dp"  
139         android:text="VT Mensal: R$ 0,00"  
140         android:textAppearance="@style/TextAppearance.AppCompat.Body2"  
141         android:textColor="#000000"  
142         android:textSize="20sp" />
```

**Figura 36 – Anexo 14 - Representação em Código XML do Layout do App Android [6/6]**

```
123     <TextView  
124         android:id="@+id/textViewVR"  
125         android:layout_width="match_parent"  
126         android:layout_height="wrap_content"  
127         android:layout_marginTop="10dp"  
128         android:layout_marginBottom="20dp"  
129         android:text="VR Mensal: R$ 0,00"  
130         android:textAppearance="@style/TextAppearance.AppCompat.Body2"  
131         android:textColor="#000000"  
132         android:textSize="20sp" />  
133  
134     <TextView  
135         android:id="@+id/textViewVT"  
136         android:layout_width="match_parent"  
137         android:layout_height="wrap_content"  
138         android:layout_marginTop="10dp"  
139         android:text="VT Mensal: R$ 0,00"  
140         android:textAppearance="@style/TextAppearance.AppCompat.Body2"  
141         android:textColor="#000000"  
142         android:textSize="20sp" />  
143     </LinearLayout>  
144  
145     </LinearLayout>  
146  
147 </LinearLayout>  
148
```

Figura 37 – Anexo 15 - Código de execução em Java do App Android [1/3]

```
1 package com.example.teste2;
2 import android.os.Bundle;
3 import android.view.View;
4 import android.widget.EditText;
5 import android.widget.TextView;
6 import android.widget.CheckBox;
7 import androidx.appcompat.app.AppCompatActivity;
8 import java.text.DecimalFormat;
9
10 public class MainActivity extends AppCompatActivity {
11     private EditText editTextSal;
12     private EditText editTextDias;
13     private EditText editTextVR;
14     private TextView textViewLiq;
15     private TextView textViewINSS;
16     private TextView textViewVR;
17     private TextView textViewVT;
18     private TextView textViewError;
19     private CheckBox checkBoxVT;
20
21     DecimalFormat df = new DecimalFormat( pattern: "#.##" );
22 }
```

**Figura 38 – Anexo 16 - Código de execução em Java do App Android [2/3]**

```
24
25     @Override
26     protected void onCreate(Bundle savedInstanceState) {
27         super.onCreate(savedInstanceState);
28         setContentView(R.layout.activity_main);
29         editTextSal = findViewById(R.id.editTextSal);
30         editTextDias = findViewById(R.id.editTextDias);
31         editTextVR = findViewById(R.id.editTextVR);
32         textViewLiq = findViewById(R.id.textViewLiq);
33         textViewINSS = findViewById(R.id.textViewINSS);
34         textViewVR = findViewById(R.id.textViewVR);
35         textViewVT = findViewById(R.id.textViewVT);
36         textViewErro = findViewById(R.id.textViewErro);
37         checkBoxVT = findViewById(R.id.checkBoxVT);
38
39
40
41     public void calcularFolha(View view) {
42         try {
43             double salBruto = Double.parseDouble(editTextSal.getText().toString());
44             double vrDia = Double.parseDouble(editTextVR.getText().toString());
45             int diasTrab = Integer.parseInt(editTextDias.getText().toString());
46             calculoSalario(salBruto);
47             calcularVR(diasTrab, vrDia);
48         } catch (NumberFormatException e) {
49             textViewErro.setVisibility(View.VISIBLE);
50         }
51     }
52 }
```

Figura 39 – Anexo 17 - Código de execução em Java do App Android [3/3]

```
53     public void calcularVR(int diasTrab, double vrDia) {  
54         double vrTotal = vrDia * diasTrab;  
55         textViewVR.setText("VR Mensal: R$ " + df.format(vrTotal));  
56     }  
57  
58     1 usage  
59     public void calculoSalario(double salBruto) {  
60         double inss = 0;  
61         double salLiquido = 0;  
62         double vt = 0;  
63  
64             if (salBruto < 1302) {  
65                 inss = salBruto * 0.075;  
66                 salLiquido = salBruto - inss;  
67             } else if (salBruto < 2571.29) {  
68                 inss = salBruto * 0.09;  
69                 salLiquido = salBruto - inss;  
70             } else if (salBruto < 3856.94) {  
71                 inss = salBruto * 0.12;  
72                 salLiquido = salBruto - inss;  
73             } else {  
74                 inss = salBruto * 0.14;  
75                 salLiquido = salBruto - inss;  
76             }  
77  
78             if (checkBoxVT.isChecked()) {  
79                 vt = salLiquido * 0.06;  
80                 salLiquido = salLiquido - vt;  
81                 textViewVT.setText("VT Mensal: R$ " + df.format(vt));  
82             }  
83             textViewINSS.setText("Desconto INSS: R$ " + df.format(inss));  
84             textViewLiq.setText("Salário Líquido: R$ " + df.format(salLiquido));  
85     }
```

**Figura 40 – Anexo 18 - Objeto comunicador com o Banco de Dados [1/8]**

```
namespace PIMteste
{
    public class FuncBD
    {
        public SqlCommand cmd = new SqlCommand();
        public String sqlcon = @"Data Source=DESKTOP-TA6HC6V\SQLEXPRESS;Initial Catalog=dbPIM;Integrated Security=True";

        public bool cadastrarDB(Func c)
        {
            cmd.CommandText =
                "insert into tbFuncionario(FuncNome, FuncCPF, FuncNasc, FuncReg, FuncCargo, FuncSal) values (@nome, @cpf, " +
                "@nasc, @reg, @cargo, @salario)";
            cmd.Parameters.AddWithValue(parameterName: "@nome", c.Nome);
            cmd.Parameters.AddWithValue(parameterName: "@cargo", c.Cargo);
            cmd.Parameters.AddWithValue(parameterName: "@salario", c.Salario);
            cmd.Parameters.AddWithValue(parameterName: "@nasc", c.Nasc);
            cmd.Parameters.AddWithValue(parameterName: "@cpf", c.Cpf);
            cmd.Parameters.AddWithValue(parameterName: "@reg", c.Reg);

            SqlConnection con = new SqlConnection(sqlcon);
            try
            {
                con.Open();
                cmd.Connection = con;
                cmd.ExecuteNonQuery();
                con.Close();
                return true;
            }
            catch (SqlException ex)
```

**Figura 41 – Anexo 19 - Objeto comunicador com o Banco de Dados [2/8]**

```
        {
            string erro = ex.Message;
            MessageBox.Show(erro);
            return false;
        }
        finally
        {
            con.Close();
        }
    }

    public List<Func> listar()
    {
        List<Func> lista = new List<Func>();
        Func c = new Func();

        cmd.CommandText =
            "select FuncID, FuncNome, FuncCPF, FuncNasc, FuncReg, FuncCargo, FuncSal from tbFuncionario";
        SqlConnection con = new SqlConnection(sqlcon);

        try
        {
            //abre o BD
            con.Open();
            cmd.Connection = con;
            //Executa query sql
            SqlDataReader dataReader = cmd.ExecuteReader();
            while (dataReader.Read())
```

**Figura 42 – Anexo 20 - Objeto comunicador com o Banco de Dados [3/8]**

The screenshot shows a code editor window with the file 'ClienteBD.cs' open. The code is part of a class named 'PIMteste'. It contains a method 'buscarFuncionario' which reads data from a database using a SqlDataReader. The code retrieves columns: Id, Nome, CPF, Nasc, Reg, Cargo, and Sal. It then adds these values to a list of 'Func' objects. If an exception occurs, it catches it and displays an error message. Finally, it closes the connection. The code editor interface includes tabs for 'cmd' and 'cmd', status bars at the bottom, and a vertical scroll bar on the right.

```
    while (dataReader.Read())
    {
        int Id = dataReader.GetInt32(0);
        string Nome = dataReader.GetString(1);
        string CPF = dataReader.GetString(2);
        string Nasc = dataReader.GetString(3);
        string Reg = dataReader.GetString(4);
        string Cargo = dataReader.GetString(5);
        double Sal = dataReader.GetDouble(6);
        lista.Add(item: new Func(Id, Nome, CPF, Nasc, Reg, Cargo, Sal));
    }

    con.Close();
}
catch (SqlException ex)
{
    string erro = ex.Message;
    MessageBox.Show(erro);
}
finally
{
    con.Close();
}

return lista;
}

referencia
public Func buscar(string nome)
{
    Func c = new Func();
```

**Figura 43 – Anexo 21 - Objeto comunicador com o Banco de Dados [4/8]**

The screenshot shows a code editor window with the file 'ClienteBD.cs' open. The code is part of a class named 'PIMteste'. It contains a method 'buscarFuncionario' which performs a search based on a name. It uses a SqlCommand with a WHERE clause to filter the 'tbFuncionario' table by 'FuncNome'. The cmd.Parameters.AddWithValue method is used to set the parameter value to 'nome'. A SqlConnection 'con' is used to connect to the database. A try-catch block handles the execution of the command and reading the results. If a record is found, it creates a new 'Func' object with the retrieved data. If no record is found, it creates a 'Func' object with a specific error message ('Não Encontrado'). Finally, it closes the connection. The code editor interface includes tabs for 'cmd' and 'cmd', status bars at the bottom, and a vertical scroll bar on the right.

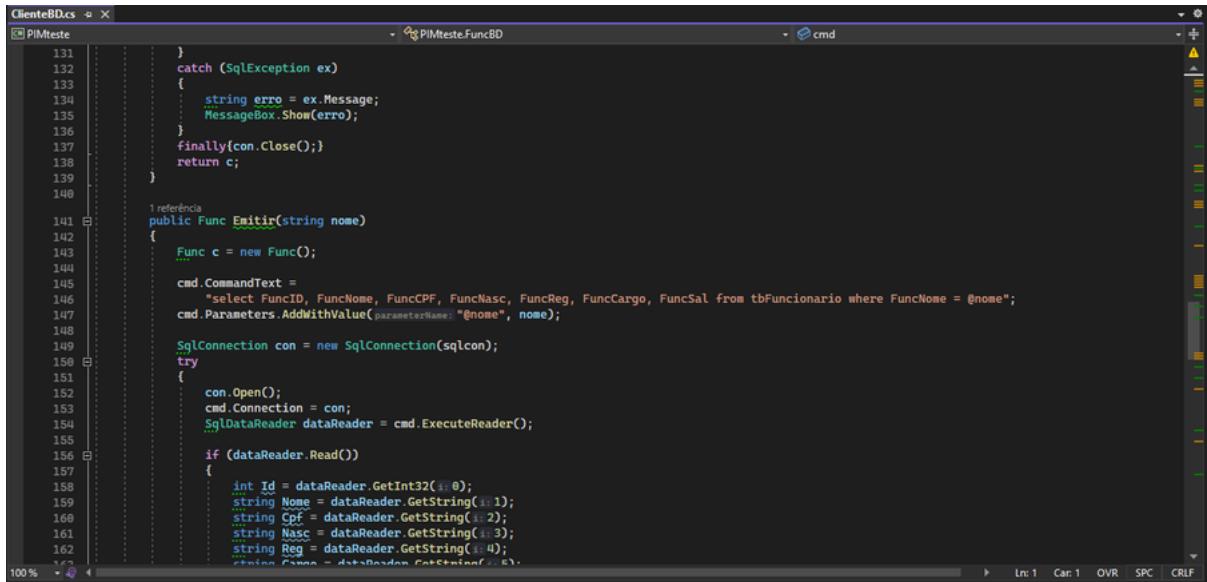
```
    Func c = new Func();

    cmd.CommandText =
        "select FuncID, FuncNome, FuncCPF, FuncNasc, FuncReg, FuncCargo, FuncSal from tbFuncionario where FuncNome = @nome";
    cmd.Parameters.AddWithValue(parameterName: "@nome", nome);

    SqlConnection con = new SqlConnection(sqlcon);
    try
    {
        con.Open();
        cmd.Connection = con;
        SqlDataReader dataReader = cmd.ExecuteReader();

        if (dataReader.Read())
        {
            int Id = dataReader.GetInt32(0);
            string Nome = dataReader.GetString(1);
            string CPF = dataReader.GetString(2);
            string Nasc = dataReader.GetString(3);
            string Reg = dataReader.GetString(4);
            string Cargo = dataReader.GetString(5);
            double Sal = dataReader.GetDouble(6);
            c = new Func(Id, Nome, CPF, Nasc, Reg, Cargo, Sal);
        }
        else
        {
            c = new Func(id: -1, nome: "Não Encontrado", cpf: "null", nasc: "null", reg: "null", cargo: "null", salario: 0);
        }
        con.Close();
    }
    catch (SqlException ex)
    {
```

Figura 44 – Anexo 22 - Objeto comunicador com o Banco de Dados [5/8]

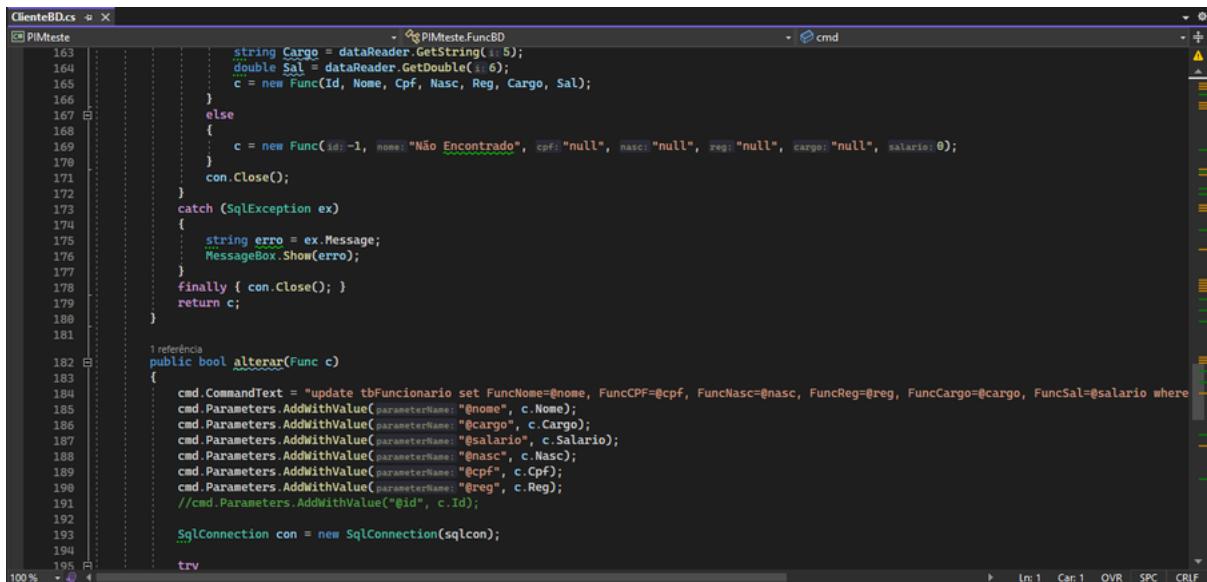


```

131     }
132     catch (SqlException ex)
133     {
134         string erro = ex.Message;
135         MessageBox.Show(erro);
136     }
137     finally { con.Close(); }
138     return c;
139 }
140
141 1 referência
142 public Func Emitir(string nome)
143 {
144     Func c = new Func();
145
146     cmd.CommandText =
147         "select FuncID, FuncNome, FuncCPF, FuncNasc, FuncReg, FuncCargo, FuncSal from tbFuncionario where FuncNome = @nome";
148     cmd.Parameters.AddWithValue(parameterName: "@nome", nome);
149
150     SqlConnection con = new SqlConnection(sqlcon);
151     try
152     {
153         con.Open();
154         cmd.Connection = con;
155         SqlDataReader dataReader = cmd.ExecuteReader();
156
157         if (dataReader.Read())
158         {
159             int Id = dataReader.GetInt32(0);
160             string Nome = dataReader.GetString(1);
161             string Cpf = dataReader.GetString(2);
162             string Nasc = dataReader.GetString(3);
163             string Reg = dataReader.GetString(4);
164             string Cargo = dataReader.GetString(5);
165             double Sal = dataReader.GetDouble(6);
166             c = new Func(Id, Nome, Cpf, Nasc, Reg, Cargo, Sal);
167         }
168         else
169         {
170             c = new Func(id: -1, nome: "Não Encontrado", cpf: "null", nasc: "null", reg: "null", cargo: "null", salario: 0);
171         }
172         con.Close();
173     }
174     catch (SqlException ex)
175     {
176         string erro = ex.Message;
177         MessageBox.Show(erro);
178     }
179     finally { con.Close(); }
180     return c;
181 }
182
183 1 referência
184 public bool alterar(Func c)
185 {
186     cmd.CommandText = "update tbFuncionario set FuncNome=@nome, FuncCPF=@cpf, FuncNasc=@nasc, FuncReg=@reg, FuncCargo=@cargo, FuncSal=@salario where";
187     cmd.Parameters.AddWithValue(parameterName: "@nome", c.Nome);
188     cmd.Parameters.AddWithValue(parameterName: "@cargo", c.Cargo);
189     cmd.Parameters.AddWithValue(parameterName: "@salario", c.Salario);
190     cmd.Parameters.AddWithValue(parameterName: "@nasc", c.Nasc);
191     cmd.Parameters.AddWithValue(parameterName: "@cpf", c.Cpf);
192     cmd.Parameters.AddWithValue(parameterName: "@reg", c.Reg);
193     //cmd.Parameters.AddWithValue("@id", c.Id);
194
195     SqlConnection con = new SqlConnection(sqlcon);
196
197     try
198     {
199         cmd.ExecuteNonQuery();
200     }
201     catch (SqlException ex)
202     {
203         string erro = ex.Message;
204         MessageBox.Show(erro);
205     }
206     finally { con.Close(); }
207     return true;
208 }

```

Figura 45 – Anexo 23 - Objeto comunicador com o Banco de Dados [6/8]



```

163     string Cargo = dataReader.GetString(5);
164     double Sal = dataReader.GetDouble(6);
165     c = new Func(Id, Nome, Cpf, Nasc, Reg, Cargo, Sal);
166 }
167 else
168 {
169     c = new Func(id: -1, nome: "Não Encontrado", cpf: "null", nasc: "null", reg: "null", cargo: "null", salario: 0);
170 }
171 con.Close();
172
173 catch (SqlException ex)
174 {
175     string erro = ex.Message;
176     MessageBox.Show(erro);
177 }
178 finally { con.Close(); }
179 return c;
180 }

182 1 referência
183 public bool alterar(Func c)
184 {
185     cmd.CommandText = "update tbFuncionario set FuncNome=@nome, FuncCPF=@cpf, FuncNasc=@nasc, FuncReg=@reg, FuncCargo=@cargo, FuncSal=@salario where";
186     cmd.Parameters.AddWithValue(parameterName: "@nome", c.Nome);
187     cmd.Parameters.AddWithValue(parameterName: "@cargo", c.Cargo);
188     cmd.Parameters.AddWithValue(parameterName: "@salario", c.Salario);
189     cmd.Parameters.AddWithValue(parameterName: "@nasc", c.Nasc);
190     cmd.Parameters.AddWithValue(parameterName: "@cpf", c.Cpf);
191     cmd.Parameters.AddWithValue(parameterName: "@reg", c.Reg);
192     //cmd.Parameters.AddWithValue("@id", c.Id);
193
194     SqlConnection con = new SqlConnection(sqlcon);
195
196     try
197     {
198         cmd.ExecuteNonQuery();
199     }
200     catch (SqlException ex)
201     {
202         string erro = ex.Message;
203         MessageBox.Show(erro);
204     }
205     finally { con.Close(); }
206     return true;
207 }

```

**Figura 46 – Anexo 24 - Objeto comunicador com o Banco de Dados [7/8]**

The screenshot shows a code editor window with the following details:

- Title Bar:** ClienteBD.cs
- Project Explorer:** PIMteste
- Code Editor:** Contains C# code for a class named FuncBD. The code includes methods for inserting, updating, and deleting data from a database table named tbFuncionario.
- Toolbars:** Standard .NET development toolbar with icons for file operations, search, and navigation.
- Status Bar:** Shows file path (PIMteste.FuncBD), line count (Ln: 1), character count (Car: 1), and other development status indicators.

```
192     SqlConnection con = new SqlConnection(sqlcon);
193
194     try
195     {
196         con.Open();
197         cmd.Connection = con;
198         cmd.ExecuteNonQuery();
199         con.Close();
200         return true;
201     }
202     catch (SqlException ex)
203     {
204         string erro = ex.Message;
205         MessageBox.Show(erro);
206         return false;
207     }
208     finally{con.Close();}
209
210
211     1 referência
212     public bool excluir(string nome)
213     {
214         cmd.CommandText = "Delete from tbFuncionario where FuncNome = @nome";
215         cmd.Parameters.AddWithValue(parameterName: "@nome", nome);
216
217         SqlConnection con = new SqlConnection(sqlcon);
218         try
219         {
220             con.Open();
221             cmd.Connection = con;
222
223             cmd.ExecuteNonQuery();
224
225             con.Close();
226             return true;
227         }
228         catch (SqlException ex)
229         {
230             string erro = ex.Message;
231             MessageBox.Show(erro);
232             return false;
233         }
234     }
235     }
236     }
237 }
```

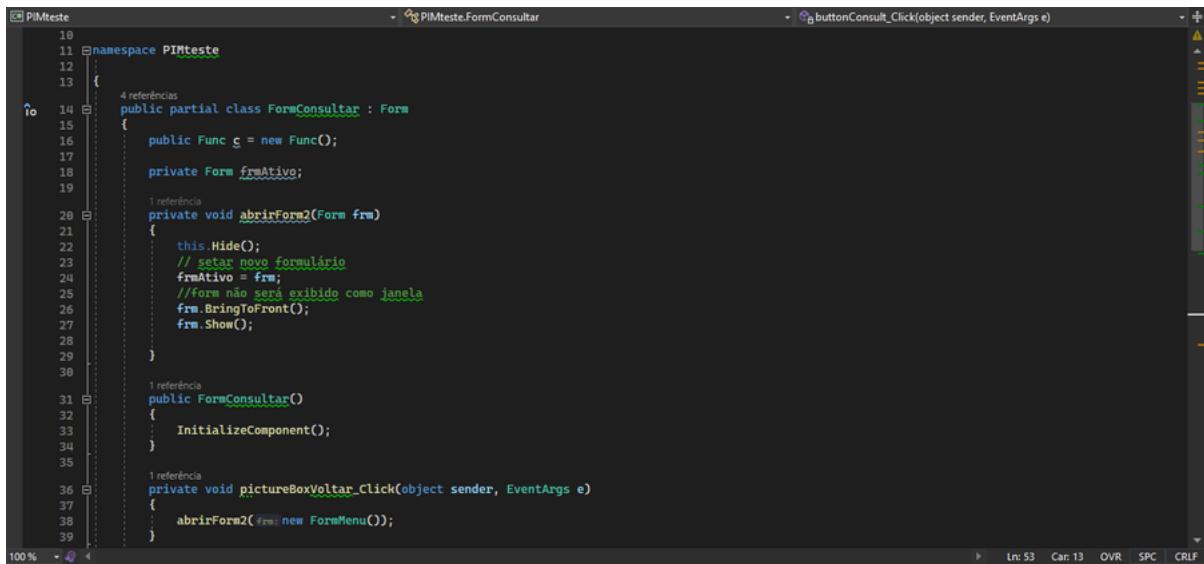
**Figura 47 – Anexo 25 - Objeto comunicador com o Banco de Dados [8/8]**

The screenshot shows a code editor window with the following details:

- Title Bar:** ClienteBD.cs
- Project Explorer:** PIMteste
- Code Editor:** Contains C# code for a class named FuncBD. The code includes methods for inserting, updating, and deleting data from a database table named tbFuncionario.
- Toolbars:** Standard .NET development toolbar with icons for file operations, search, and navigation.
- Status Bar:** Shows file path (PIMteste.FuncBD), line count (Ln: 1), character count (Car: 1), and other development status indicators.

```
216     SqlConnection con = new SqlConnection(sqlcon);
217     try
218     {
219         con.Open();
220         cmd.Connection = con;
221
222         cmd.ExecuteNonQuery();
223
224         con.Close();
225         return true;
226     }
227     catch (SqlException ex)
228     {
229         string erro = ex.Message;
230         MessageBox.Show(erro);
231         return false;
232     }
233     finally{con.Close();}
234
235     }
236     }
237 }
```

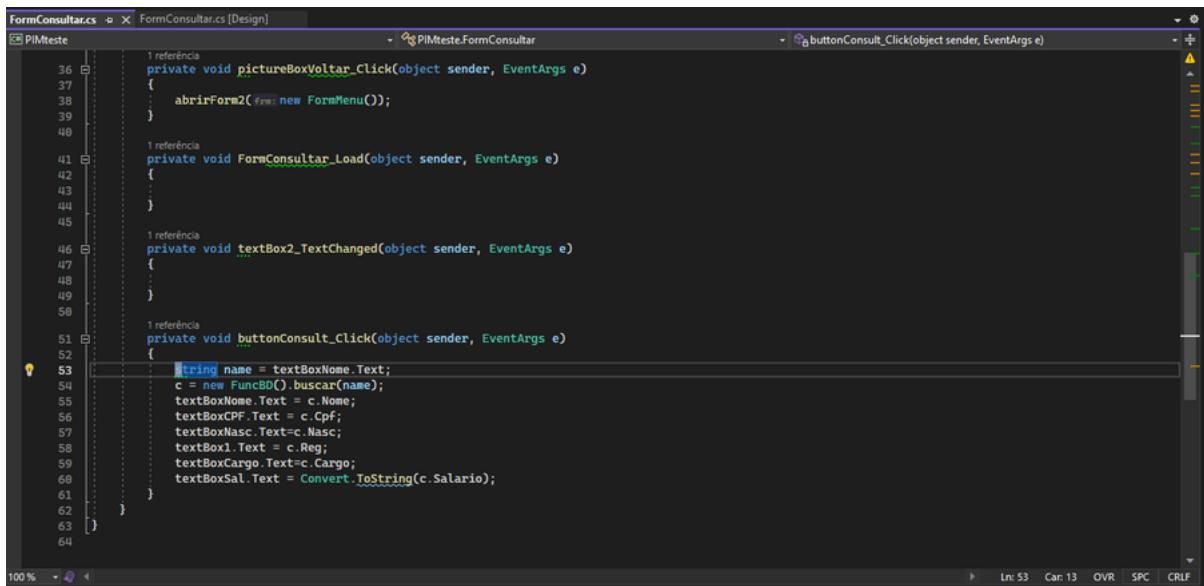
Figura 48 – Anexo 26 - Objeto relacionado a função de consulta de cadastro [1/2]



```
10
11  namespace PIMteste
12  {
13  }
14  public partial class FormConsultar : Form
15  {
16      public Func c = new Func();
17
18      private Form frmAtivo;
19
20      private void abrirForm2(Form frm)
21      {
22          this.Hide();
23          // setar novo formulário
24          frmAtivo = frm;
25          //form não será exibido como janela
26          frm.BringToFront();
27          frm.Show();
28      }
29
30
31  public FormConsultar()
32  {
33      InitializeComponent();
34  }
35
36  private void pictureBoxVoltar_Click(object sender, EventArgs e)
37  {
38      abrirForm2(new FormMenu());
39  }

```

Figura 49 – Anexo 27 - Objeto relacionado a função de consulta de cadastros [2/2]



```
36  private void pictureBoxVoltar_Click(object sender, EventArgs e)
37  {
38      abrirForm2(new FormMenu());
39  }
40
41  private void FormConsultar_Load(object sender, EventArgs e)
42  {
43  }
44
45
46  private void textBox2_TextChanged(object sender, EventArgs e)
47  {
48  }
49
50
51  private void buttonConsult_Click(object sender, EventArgs e)
52  {
53      string name = textBoxNome.Text;
54      c = new FuncBD().buscar(name);
55      textBoxNome.Text = c.Nome;
56      textBoxCPF.Text = c.Cpf;
57      textBoxNasc.Text = c.Nasc;
58      textBox1.Text = c.Reg;
59      textBoxCargo.Text = c.Cargo;
60      textBoxSal.Text = Convert.ToString(c.Salario);
61  }
62
63  }
64
```

**Figura 50 – Anexo 28 - Objeto relacionado a função de edição de cadastros [1/2]**

The screenshot shows the Visual Studio IDE with the FormEditor.cs file open. The code defines a partial class FormEditor that inherits from Form. It includes methods for opening new forms, initializing components, and handling form load and button click events. The code is annotated with several 'referência' (reference) comments.

```
13
14     4 referências
15     public partial class FormEditor : Form
16     {
17
18         private Form frmAtivo;
19
20         public List<Func> lista = new List<Func>();
21
22         1 referência
23         private void abrirForm2(Form frm)
24         {
25             this.Hide();
26             // setar novo formulário
27             frmAtivo = frm;
28             //form não será exibido como janela
29             frm.BringToFront();
30             frm.Show();
31
32         1 referência
33         public FormEditor()
34         {
35             InitializeComponent();
36
37         1 referência
38         private void FormEditor_Load(object sender, EventArgs e)
39         {
40
41         1 referência
42         private void pictureBoxVoltar_Click(object sender, EventArgs e)
43         {
44
45
46         1 referência
47         private void buttonEditar_Click(object sender, EventArgs e)
48         {
49             string nome, cpf, nasc, reg, cargo;
50             double sal;
51             nome=textBoxNome.Text;
52             cpf=textBoxCPF.Text;
53             nasc(textBoxNasc.Text);
54             reg=textBox1.Text;
55             cargo=textBoxCargo.Text;
56             sal=Convert.ToDouble(textBoxSal.Text);
57             bool ok = new FuncBD().alterar(<new Func(nome, cpf, nasc, reg, cargo, sal));
58             if (ok)
59             {
60                 lista = new FuncBD().Listar();
61                 MessageBox.Show(text: "Dados Alterados com Sucesso!");
62             }
63             else
64             {
65                 MessageBox.Show(text: "Erro ao Tentar Alterar Cadastro");
66             }
67
68         }
69     }
70 }
```

**Figura 51 – Anexo 29 - Objeto relacionado a função de edição de cadastros [2/2]**

The screenshot shows the Visual Studio IDE with the FormEditor.cs file open. This part of the code handles the buttonEdit\_Click event. It retrieves data from text boxes, converts salary to a double, and calls a FuncBD method to update the database. It then displays a success or error message box. The code is annotated with several 'referência' (reference) comments.

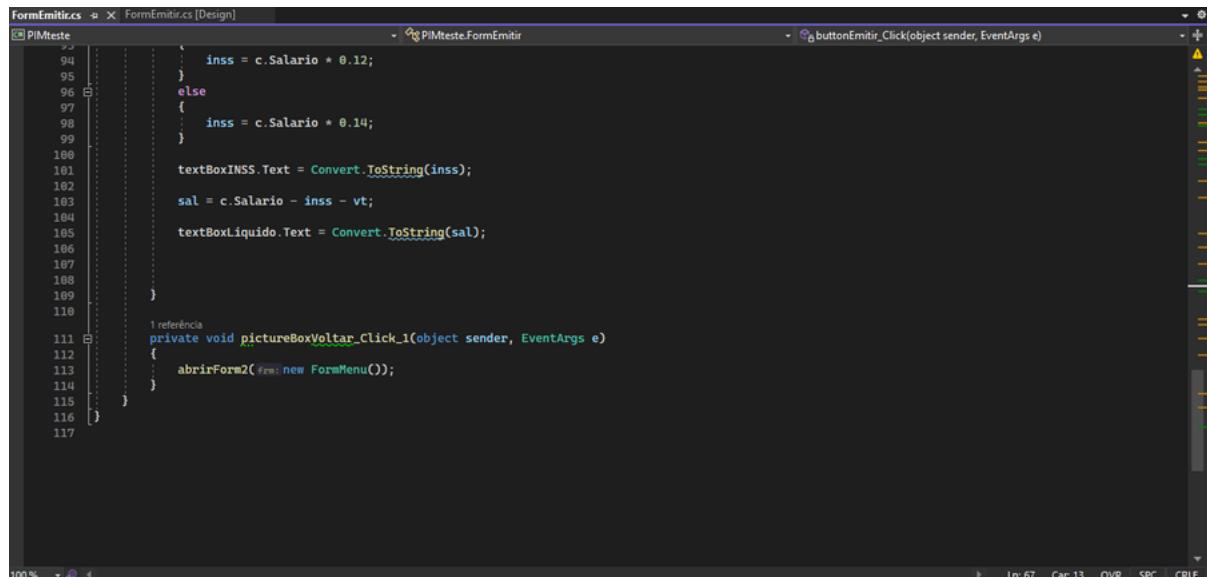
```
41
42         1 referência
43         private void pictureBoxVoltar_Click(object sender, EventArgs e)
44         {
45             abrirForm2(frm: new FormMenu());
46
47         1 referência
48         private void buttonEditar_Click(object sender, EventArgs e)
49         {
50             string nome, cpf, nasc, reg, cargo;
51             double sal;
52             nome=textBoxNome.Text;
53             cpf=textBoxCPF.Text;
54             nasc(textBoxNasc.Text);
55             reg=textBox1.Text;
56             cargo=textBoxCargo.Text;
57             sal=Convert.ToDouble(textBoxSal.Text);
58             bool ok = new FuncBD().alterar(<new Func(nome, cpf, nasc, reg, cargo, sal));
59             if (ok)
60             {
61                 lista = new FuncBD().Listar();
62                 MessageBox.Show(text: "Dados Alterados com Sucesso!");
63             }
64             else
65             {
66                 MessageBox.Show(text: "Erro ao Tentar Alterar Cadastro");
67             }
68         }
69     }
70 }
```

**Figura 52 – Anexo 30 - Objeto relacionado a função de emitir a folha de pagamento [1/3]**

```
FormEmitir.cs  FormEmitir.cs [Design]  PIMteste.FormEmitir  buttonEmitir_Click(object sender, EventArgs e)
11  namespace PIMteste
12  {
13      public partial class FormEmitir : Form
14      {
15          public Func c = new Func();
16          public FormEmitir()
17          {
18              InitializeComponent();
19          }
20
21          private Form frmAtivo;
22
23          private void abrirForm2(Form frm)
24          {
25              this.Hide();
26              // setar novo formulário
27              frmAtivo = frm;
28              //form não será exibido como janela
29              frm.BringToFront();
30              frm.Show();
31
32          }
33
34          private void label1_Click(object sender, EventArgs e)
35          {
36
37          }
38
39          private void tableCads_Paint(object sender, PaintEventArgs e)
40          {
41
42      }
}
100%  Lr: 67  Car: 13  OVR  SPC  CRLF
```

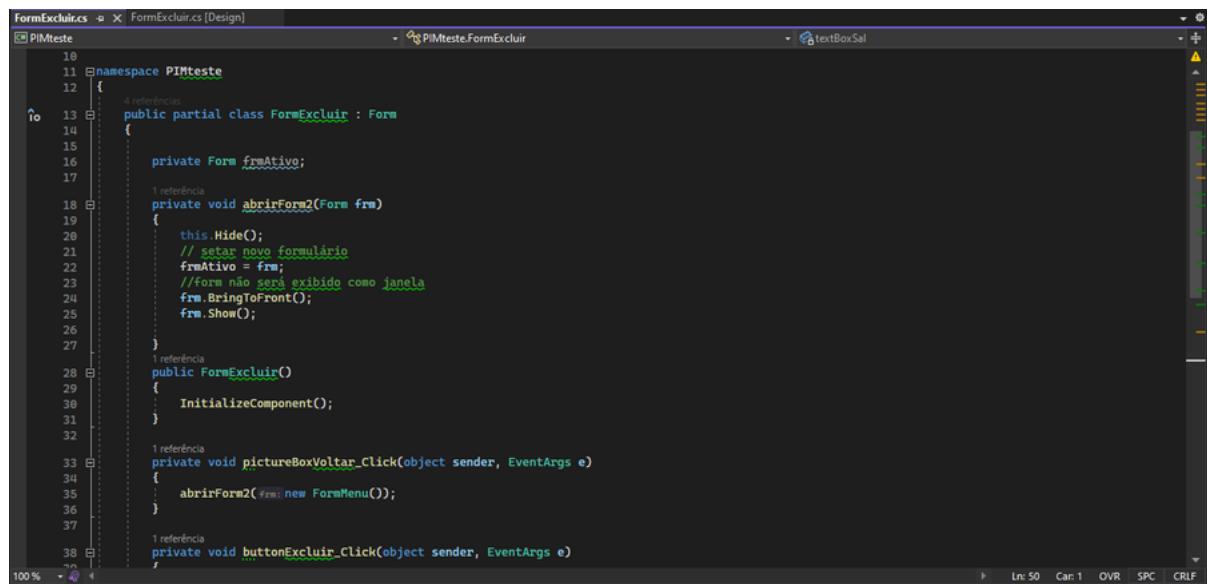
**Figura 53 – Anexo 31 - Objeto relacionado a função de emitir a folha de pagamento [2/3]**

```
FormEmitir.cs  FormEmitir.cs [Design]  PIMteste.FormEmitir  buttonEmitir_Click(object sender, EventArgs e)
64
65  private void buttonEmitir_Click(object sender, EventArgs e)
66  {
67      labelNomeEx.Hide();
68
69      double sal, inss, vt;
70      string name = textBoxNome.Text;
71      c = new FuncBD().Emitir(name);
72      textBoxNome.Text = c.Nome;
73      textBoxCPF.Text = c.Cpf;
74      textBoxNasc.Text = c.Nasc;
75      textBox1.Text = c.Reg;
76      textBoxCargo.Text = c.Cargo;
77      textBoxSal.Text = Convert.ToString(c.Salario);
78
79      vt = c.Salario * 0.06;
80      textBoxVT.Text = Convert.ToString(vt);
81
82      if (c.Salario < 1302)
83      {
84          inss = c.Salario * 0.075;
85      }
86
87      if (c.Salario < 2571.29)
88      {
89          inss = c.Salario * 0.009;
90      }
91
92      if (c.Salario < 3856.94)
93      {
94          inss = c.Salario * 0.12;
95      }
96  else
97
98
99
100%  Lr: 67  Car: 13  OVR  SPC  CRLF
```

**Figura 54 – Anexo 32 - Objeto relacionado a função de emitir a folha de pagamento [3/3]**

```
FormEmitir.cs  FormEmitir.cs [Design]  PIMteste.FormEmitir  buttonEmitir_Click(object sender, EventArgs e)

94     if (c.Salario > 1200)
95     {
96         inss = c.Salario * 0.12;
97     }
98     else
99     {
100        inss = c.Salario * 0.14;
101    }
102
103    textBoxINSS.Text = Convert.ToString(inss);
104
105    sal = c.Salario - inss - vt;
106
107    textBoxLiquido.Text = Convert.ToString(sal);
108
109}
110
111// referência
112private void pictureBoxVoltar_Click_1(object sender, EventArgs e)
113{
114    abrirForm2(frm: new FormMenu());
115}
116
117
```

**Figura 55 – Anexo 33 - Objeto relacionado a função de exclusão de cadastro [1/2]**

```
FormExcluir.cs  FormExcluir.cs [Design]  PIMteste.FormExcluir  textBoxSal

10
11  namespace PIMteste
12  {
13      // referência
14      public partial class FormExcluir : Form
15      {
16
17          private Form frmAtivo;
18
19          // referência
20          private void abrirForm2(Form frm)
21          {
22              this.Hide();
23              // setar novo formulário
24              frmAtivo = frm;
25              //form não será exibido como janela
26              frm.BringToFront();
27              frm.Show();
28
29          }
30
31          // referência
32          public void Excluir()
33          {
34              InitializeComponent();
35
36          }
37
38          // referência
39          private void pictureBoxVoltar_Click(object sender, EventArgs e)
40          {
41              abrirForm2(frm: new FormMenu());
42
43          }
44
45          // referência
46          private void buttonExcluir_Click(object sender, EventArgs e)
47          {
48
49          }
50
51      }
52  }
```

**Figura 56 – Anexo 34 - Objeto relacionado a função de exclusão de cadastro [2/2]**

```
FormExcluir.cs [Design] PIMteste.FormExcluir textBoxSal

27 }
28 {
29     InitializeComponent();
30 }
31 }
32 }
33 {
34     abrirForm2(frm: new FormMenu());
35 }
36 }
37 }
38 {
39     string nome;
40     nome = textBoxNome.Text;
41     bool ok = new FuncBD().excluir(nome);
42     if (ok)
43     {
44         MessageBox.Show(text: "Cadastro Excluido com Sucesso");
45     }
46 }
47 }
48 }
49 }
50 }
```

The screenshot shows the code editor with the file `FormExcluir.cs` open. The code is written in C# and defines a class `FormExcluir`. It contains several methods and event handlers. One method, `buttonExcluir_Click`, is shown in detail, where it retrieves a name from a text box, calls a database function to delete the record, and then displays a success message box.

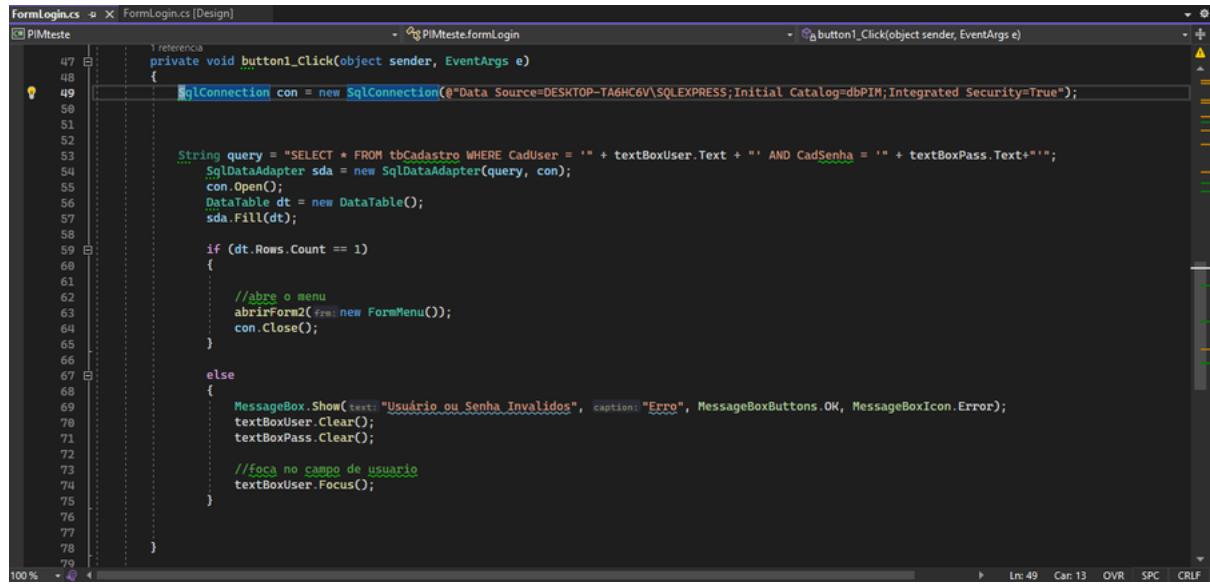
**Figura 57 – Anexo 35 - Objeto relacionado a função de Login do sistema [1/2]**

```
FormLogin.cs [Design] PIMteste.formLogin button1_Click

10 using System.Data.SqlClient;
11 [using static System.Windows.Forms.VisualStyles.VisualStyleElement;
12 
13 namespace PIMteste
14 {
15     3 referências
16     public partial class formLogin : Form
17     {
18         private Form frmAtivo;
19 
20         1 referência
21         public formLogin()
22         {
23             InitializeComponent();
24         }
25 
26         1 referência
27         private void abrirForm2(Form frm)
28         {
29             this.Hide();
30             // setar novo formulário
31             frmAtivo = frm;
32             //form não será exibido como janela
33             frm.BringToFront();
34             frm.Show();
35         }
36 
37         1 referência
38         private void label1_Click(object sender, EventArgs e)
39         {
```

The screenshot shows the code editor with the file `FormLogin.cs` open. The code is written in C# and defines a class `formLogin`. It includes a constructor and a method `abrirForm2` which hides the current form and sets a new active form. It also handles a click event for a label, though the event handler code is currently empty.

Figura 58 – Anexo 36 - Objeto relacionado a função de Login do sistema [2/2]



```

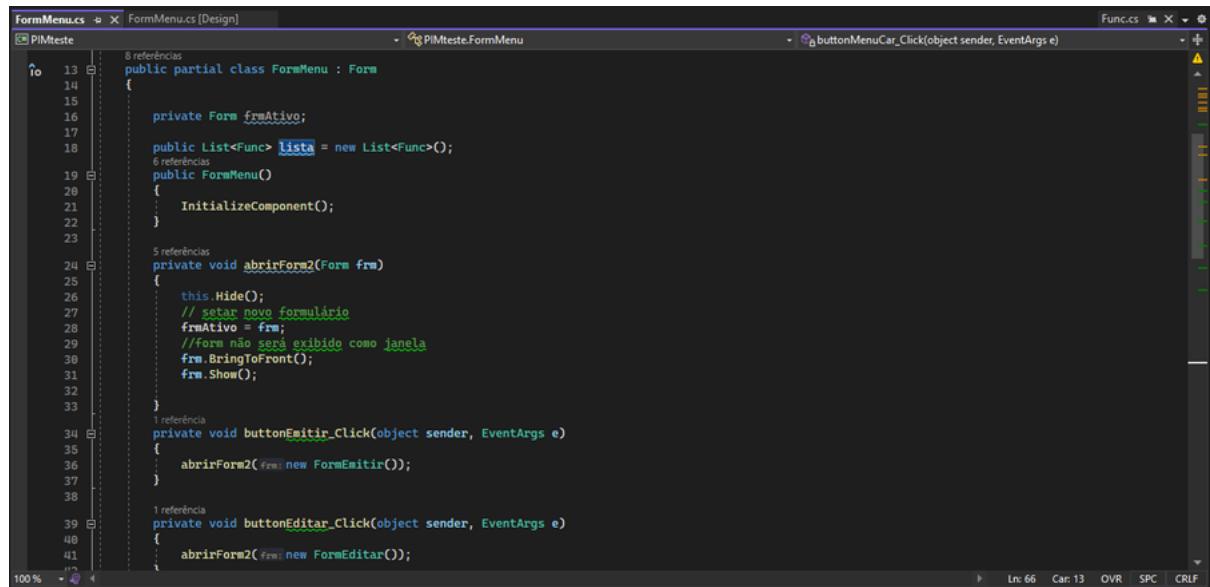
FormLogin.cs  FormLogin.cs [Design]  PIMteste  button1_Click(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection(@"Data Source=DESKTOP-TA6HC6V\SQLEXPRESS;Initial Catalog=dbPIM;Integrated Security=True");

    String query = "SELECT * FROM tbCadastro WHERE CadUser = '" + textBoxUser.Text + "' AND CadSenha = '" + textBoxPass.Text + "'";
    SqlDataAdapter sda = new SqlDataAdapter(query, con);
    con.Open();
    DataTable dt = new DataTable();
    sda.Fill(dt);

    if (dt.Rows.Count == 1)
    {
        //abre o menu
        abrirForm2(frm: new FormMenu());
        con.Close();
    }
    else
    {
        MessageBox.Show(text: "Usuário ou Senha Invalidos", caption: "Erro", MessageBoxButtons.OK, MessageBoxIcon.Error);
        textBoxUser.Clear();
        textBoxPass.Clear();
        //foca no campo de usuário
        textBoxUser.Focus();
    }
}

```

Figura 59 – Anexo 37 - Objeto relacionado as funções do menu principal [1/3]



```

FormMenu.cs  FormMenu.cs [Design]  PIMteste  FormMenu.cs  buttonMenuCar_Click(object sender, EventArgs e)
public partial class FormMenu : Form
{
    private Form frmAtivo;
    public List<Func> lista = new List<Func>();
    public FormMenu()
    {
        InitializeComponent();
    }

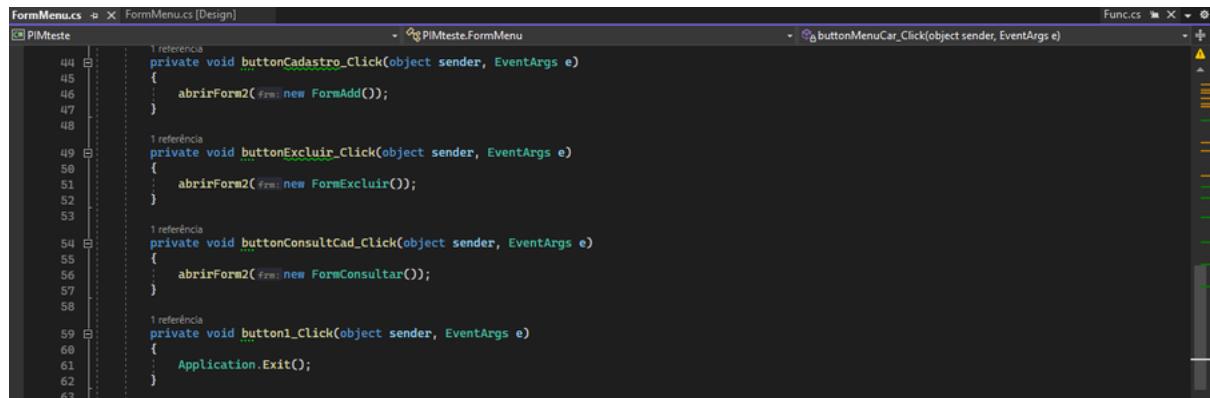
    private void abrirForm2(Form frm)
    {
        this.Hide();
        // setar novo formulário
        frmAtivo = frm;
        //form não será exibido como janela
        frm.BringToFront();
        frm.Show();
    }

    private void buttonEmitir_Click(object sender, EventArgs e)
    {
        abrirForm2(frm: new FormEmitir());
    }

    private void buttonEditar_Click(object sender, EventArgs e)
    {
        abrirForm2(frm: new FormEditar());
    }
}

```

Figura 60 – Anexo 38 - Objeto relacionado as funções do menu principal [2/3]



```

FormMenu.cs  FormMenu.cs [Design]  PIMteste  buttonMenuCar_Click(object sender, EventArgs e)
private void buttonCadastro_Click(object sender, EventArgs e)
{
    abrirForm2(frm: new FormAdd());
}

private void buttonExcluir_Click(object sender, EventArgs e)
{
    abrirForm2(frm: new FormExcluir());
}

private void buttonConsultCad_Click(object sender, EventArgs e)
{
    abrirForm2(frm: new FormConsultar());
}

private void button1_Click(object sender, EventArgs e)
{
    Application.Exit();
}

```

Figura 61 – Anexo 39 - Objeto relacionado as funções do menu principal [3/3]

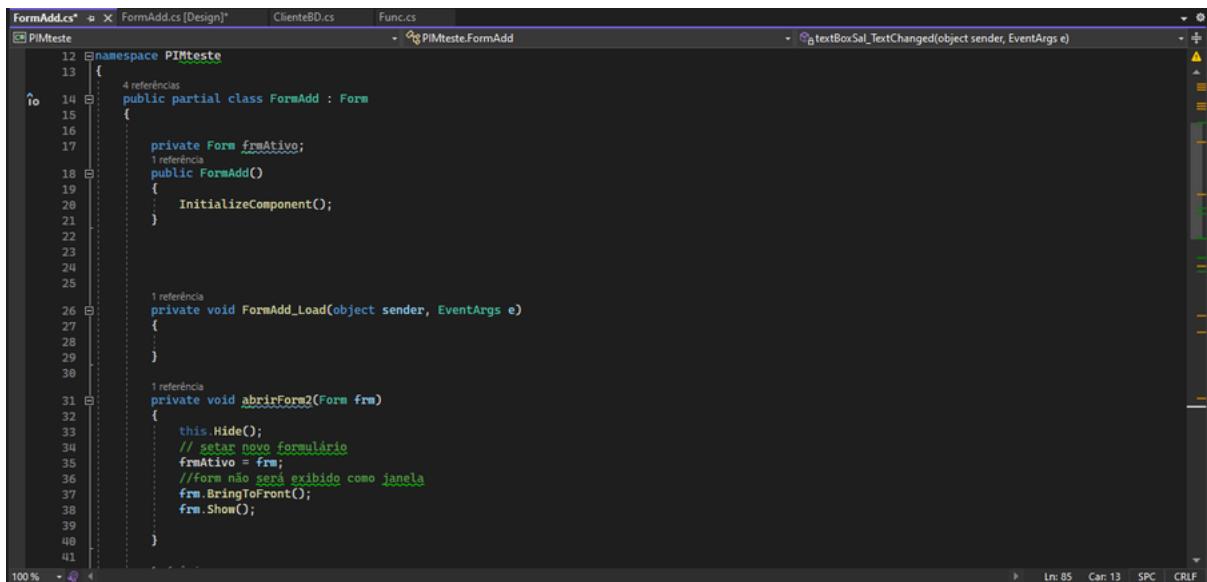


```

63
64     1 referência
65     private void buttonMenuCar_Click(object sender, EventArgs e)
66     {
67         lista = new FuncBD().listar();
68         string print = "";
69         foreach (var item in lista)
70         {
71             print += item.ToString();
72         }
73         MessageBox.Show(print);
74     }
75     }
76 }
77

```

Figura 62 – Anexo 40 - Objeto relacionado a função de adição de cadastro [1/2]

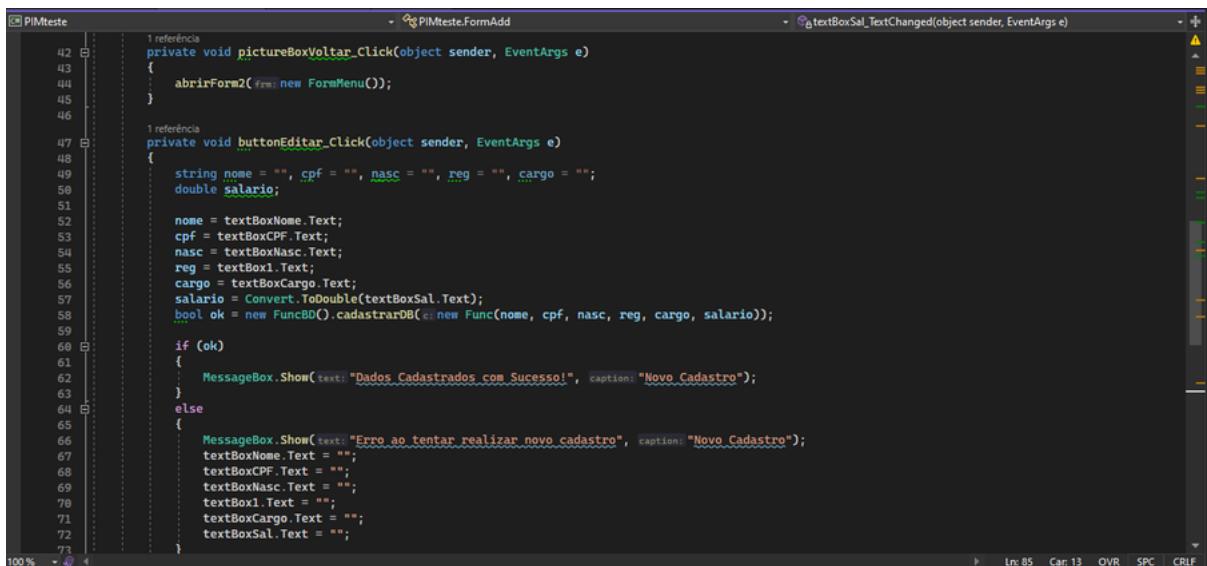


```

FormAdd.cs*  FormAdd.cs [Design]*  ClienteBD.cs  Func.cs
PIMteste
12  namespace PIMteste
13  {
14      4 referências
15      public partial class FormAdd : Form
16      {
17
18          1 referência
19          private Form frmAtivo;
20
21          public FormAdd()
22          {
23              InitializeComponent();
24
25          }
26
27          1 referência
28          private void FormAdd_Load(object sender, EventArgs e)
29          {
30
31          }
32
33          1 referência
34          private void abrirForm2(Form frm)
35          {
36              this.Hide();
37              // setar novo formulário
38              frmAtivo = frm;
39              //form não será exibido como janela
40              frm.BringToFront();
41              frm.Show();
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73

```

Figura 63 – Anexo 41 - Objeto relacionado a função de adição de cadastro [2/2]



```

PIMteste
42  1 referência
43  private void pictureBoxVoltar_Click(object sender, EventArgs e)
44  {
45      abrirForm2(frm: new FormMenu());
46  }
47
48  1 referência
49  private void buttonEditar_Click(object sender, EventArgs e)
50  {
51      string nome = "", cpf = "", nasc = "", reg = "", cargo = "";
52      double salario;
53
54      nome = textBoxNome.Text;
55      cpf = textBoxCPF.Text;
56      nasc = textBoxNasc.Text;
57      reg = textBox1.Text;
58      cargo = textBoxCargo.Text;
59      salario = Convert.ToDouble(textBoxSal.Text);
60      bool ok = new FuncBD().cadastrarDB(new Func(nome, cpf, nasc, reg, cargo, salario));
61
62      if (ok)
63      {
64          MessageBox.Show(text: "Dados Cadastrados com Sucesso!", caption: "Novo Cadastro");
65      }
66      else
67      {
68          MessageBox.Show(text: "Erro ao tentar realizar novo cadastro", caption: "Novo Cadastro");
69          textBoxNome.Text = "";
70          textBoxCPF.Text = "";
71          textBoxNasc.Text = "";
72          textBox1.Text = "";
73          textBoxCargo.Text = "";
74          textBoxSal.Text = "";
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

**Figura 64 – Anexo 42 - Objeto relacionado aos principais atributos do sistema [1/3]**

Func.cs

```
10  | 31 referências
11 | public class Func
12 |
13 |     private int id;
14 |     private string nome = "";
15 |     private string cargo = "";
16 |     private double salario;
17 |     private string nasc = "";
18 |     private string cpf = "";
19 |     private string reg = "";
20 |
21 |     2 referências
22 |     public Func(string nome, string cpf, string nasc, string reg, string cargo, double salario)
23 |     {
24 |         this.Nome = nome;
25 |         this.Cargo = cargo;
26 |         this.Salario = salario;
27 |         this.Nasc = nasc;
28 |         this.Cpf = cpf;
29 |         this.Reg = reg;
30 |
31 |         5 referências
32 |         public Func() {}
33 |
34 |         5 referências
35 |         public Func(int id, string nome, string cpf, string nasc, string reg, string cargo, double salario)
36 |         {
37 |             this.Id = id;
38 |             this.Nome = nome;
39 |             this.Cargo = cargo;
40 |             this.Salario = salario;
41 |             this.Nasc = nasc;
42 |             this.Cpf = cpf;
43 |         }
44 |
45 |         1 referência
46 |         public int Id
47 |         {
48 |             get => id;
49 |             set => id = value;
50 |
51 |             6 referências
52 |             public string Nome
53 |             {
54 |                 get => nome;
55 |                 set => nome = value;
56 |
57 |                 6 referências
58 |                 public string Cargo
59 |                 {
60 |                     get => cargo;
61 |                     set => cargo = value;
62 |
63 |                     15 referências
64 |                     public double Salario
65 |                     {
66 |                         get => salario;
67 |                         set => salario = value;
68 |                     }
69 |                 }
70 |             }
71 |         }
72 |     }
73 | }
```

**Figura 65 – Anexo 43 - Objeto relacionado aos principais atributos do sistema [2/3]**

Func.cs

```
37 |             this.Salario = salario;
38 |             this.Nasc = nasc;
39 |             this.Cpf = cpf;
40 |             this.Reg = reg;
41 |
42 |
43 |
44 |         1 referência
45 |         public int Id
46 |         {
47 |             get => id;
48 |             set => id = value;
49 |
50 |
51 |             6 referências
52 |             public string Nome
53 |             {
54 |                 get => nome;
55 |                 set => nome = value;
56 |
57 |                 6 referências
58 |                 public string Cargo
59 |                 {
60 |                     get => cargo;
61 |                     set => cargo = value;
62 |
63 |                     15 referências
64 |                     public double Salario
65 |                     {
66 |                         get => salario;
67 |                         set => salario = value;
68 |                     }
69 |                 }
70 |             }
71 |         }
72 |     }
73 | }
```

**Figura 66 – Anexo 44 - - Objeto relacionado aos principais atributos do sistema [3/3]**

```
Func.cs  PIMteste.Func  Id
67
68     public string Nasc
69     {
70         get => nasc;
71         set => nasc = value;
72     }
73
74     public string Cpf
75     {
76         get => cpf;
77         set => cpf = value;
78     }
79
80     public string Reg
81     {
82         get => reg;
83         set => reg = value;
84     }
85
86     public override string ToString()
87     {
88         return "\n Id: " + id + " Nome: " + nome + " Cargo: " + cargo + " Salário: " +
89             salario + "Data de Nascimento: "+nasc + "CPF: "+cpf +"Número de Registro: "+reg;
90     }
91 }
92
93
```

**Figura 67 – Anexo 45 - Objeto relacionado as instruções de início do sistema**

```
Program.cs  PIMteste.Program  Main()
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Threading.Tasks;
5 using System.Windows.Forms;
6
7 namespace PIMteste
8 {
9     internal static class Program
10    {
11         /// <summary>
12         /// Ponto de entrada principal para o aplicativo.
13         /// </summary>
14         [STAThread]
15         static void Main()
16         {
17             Application.EnableVisualStyles();
18             Application.SetCompatibleTextRenderingDefault(false);
19             Application.Run(new formLogin());
20         }
21     }
22 }
```