|  |  |  |
| --- | --- | --- |
|  | Ministério da Educação  **Universidade Tecnológica Federal do Paraná**  Campus Apucarana  Coordenação de Engenharia de Computação- COENC  Engenharia de Computação | utfpr |

João Pedro Cavani Meireles, RA: 2321424

Filipe Augusto Parreira Almeida, RA: 2320622

**RELATÓRIO PROJETO FINAL – PROGRAMAÇÃO ORIENTADA À OBJETO**

Apucarana

29/06/2023

1. **DESCRIÇÃO DO PROJETO**

Para o projeto final foi definido pelo professor um tema, para que fosse implementando um software que abordasse este tema. O tema escolhido pelo professo foi “prefeitura”, sendo assim, foi implementado um software para o setor de saneamento de água para uma cidade, ou seja, o setor responsável por fornecer e regulamentar o fornecimento de água.

O programa então tem como principal intuito facilitar o acesso aos serviços para o cidadão, que será o principal usuário do sistema, fornecendo serviços de auto atendimento, de certa forma que agilize o tempo do cidadão (chamado no software de Cliente), onde não será mais necessário ir até uma agência física deste setor da prefeitura pra resolver seus eventuais problemas com o serviço disponibilizado por este setor.

1. **MODELAGEM UML**

A seguir vão ser apresentados então os diagramas (e tabelas), que foram cruciais para modelar a lógica do sistema, em alguns aspectos o sistema pode não seguir exatamente a modelagem.

* Diagrama

  Descrição gerada automaticamenteDiagrama de casos de uso:

Figura . Diagrama de Casos de Uso

* Tabelas de casos de uso:

|  |  |
| --- | --- |
| Caso de Uso | Valor |
| Nome do Caso de Uso | Cadastrar |
| Atores Envolvidos | Cliente |
| Finalidade | O cliente realiza o cadastro de seus dados no sistema. |
| Descrição detalhada | O cliente insere seus dados na interface gráfica, e então é cadastrado no banco de dados. |
| Tabelas manipuladas |  |

|  |  |
| --- | --- |
| Caso de Uso | Valor |
| Nome do Caso de Uso | Cadastrar Imóvel |
| Atores Envolvidos | Cliente |
| Finalidade | O cliente cadastra um imóvel no banco de dados |
| Descrição detalhada | O cliente pode ou não cadastrar um imóvel juntamente com o cadastro, porém esse recurso também é disponibilizado posteriormente. Os imóveis cadastrados ficam vinculados ao cadastro do cliente. |
| Tabelas manipuladas |  |

|  |  |
| --- | --- |
| Caso de Uso | Valor |
| Nome do Caso de Uso | Solicitar envio de contas e-mail |
| Atores Envolvidos | Cliente |
| Finalidade | O cliente solicita o envio de suas faturas por e-mail |
| Descrição detalhada | É uma opção que o cliente realiza na hora do cadastro, onde ele solicita que suas faturas mensais sejam enviadas por e-mail automaticamente. |
| Tabelas manipuladas |  |

|  |  |
| --- | --- |
| Caso de Uso | Valor |
| Nome do Caso de Uso | Solicitar Débito Automático |
| Atores Envolvidos | Cliente |
| Finalidade | Cliente solicita que suas faturas sejam debitadas automaticamente em sua conta bancária. |
| Descrição detalhada | É uma opção onde o cliente a seleciona e insere os dados da sua conta bancária, de forma que suas faturas sejam debitadas automaticamente na conta bancária cadastrada. |
| Tabelas manipuladas |  |

|  |  |
| --- | --- |
| Caso de Uso | Valor |
| Nome do Caso de Uso | Solicitar Segunda Via de Conta |
| Atores Envolvidos | Cliente |
| Finalidade | Cliente solicita a segunda via da sua fatura mensal. |
| Descrição detalhada | Caso haja algum problema e o cliente não tenha recebido sua fatura, o mesmo, pode então solicitar a segunda via. |
| Tabelas manipuladas |  |

|  |  |
| --- | --- |
| Caso de Uso | Valor |
| Nome do Caso de Uso | Validar Cadastro |
| Atores Envolvidos | Cliente |
| Finalidade | O sistema solicita a valida o login do usuário. |
| Descrição detalhada | O sistema solicita as informações de login do usuário para que o usuário acesse sua conta e posteriormente selecione o imóvel que ele deseja solicitar a fatura. |
| Tabelas manipuladas |  |

|  |  |
| --- | --- |
| Caso de Uso | Valor |
| Nome do Caso de Uso | Selecionar Imóvel |
| Atores Envolvidos | Cliente |
| Finalidade | Permite o cliente solicitar um imóvel. |
| Descrição detalhada | O cliente pode solicitar um imóvel para visualizar as faturas relacionadas exclusivamente com o imóvel selecionado. |
| Tabelas manipuladas |  |

|  |  |
| --- | --- |
| Caso de Uso | Valor |
| Nome do Caso de Uso | Solicitar Religamento |
| Atores Envolvidos | Cliente, Funcionário |
| Finalidade | O cliente solicita o religamento do seu abastecimento de água. |
| Descrição detalhada | É enviado uma solicitação de religamento para que um funcionário possa aprovar e religar o abastecimento de água quando desligado. |
| Tabelas manipuladas |  |

|  |  |
| --- | --- |
| Caso de Uso | Valor |
| Nome do Caso de Uso | Informar Vazamento |
| Atores Envolvidos | Cliente, Funcionário |
| Finalidade | O cliente informa um vazamento para a empresa. |
| Descrição detalhada | O cliente envia uma solicitação informando um vazamento e especificando o local, para que o funcionário possa tomar as devidas providencias. |
| Tabelas manipuladas |  |

|  |  |
| --- | --- |
| Caso de Uso | Valor |
| Nome do Caso de Uso | Solicitar Atendimento |
| Atores Envolvidos | Cliente, Funcionário |
| Finalidade | O cliente solicita um eventual atendimento presencial. |
| Descrição detalhada | É realizado uma solicitação onde o cliente solicita um atendimento presencial, agendando um horário. Dessa forma o funcionário processa essa solicitação. |
| Tabelas manipuladas |  |

|  |  |
| --- | --- |
| Caso de Uso | Valor |
| Nome do Caso de Uso | Realizar Login |
| Atores Envolvidos | Cliente, Funcionário |
| Finalidade | Tanto o funcionário quanto o cliente podem realizar o login com as suas informações. |
| Descrição detalhada | É solicitado então o login do usuário (id, senha), para que seja então carregado as informações do usuário que o realiza. |
| Tabelas manipuladas |  |

|  |  |
| --- | --- |
| Caso de Uso | Valor |
| Nome do Caso de Uso | Armazenar Solicitação |
| Atores Envolvidos | Cliente, Funcionário |
| Finalidade | O sistema armazena a solicitação feita pelo cliente. |
| Descrição detalhada | A solicitação feita pelo cliente é armazenada no banco de dados possibilitando que o funcionário tenha acesso a elas. |
| Tabelas manipuladas |  |

|  |  |
| --- | --- |
| Caso de Uso | Valor |
| Nome do Caso de Uso | Verificar Solicitações |
| Atores Envolvidos | Funcionário, Cliente |
| Finalidade | Permite que o funcionário verifique as solicitações feitas pelos clientes. |
| Descrição detalhada | O funcionário pode então ter acesso as solicitações armazenadas no banco de dados, aprovando-as ou não. |
| Tabelas manipuladas |  |

* Diagrama de Classes

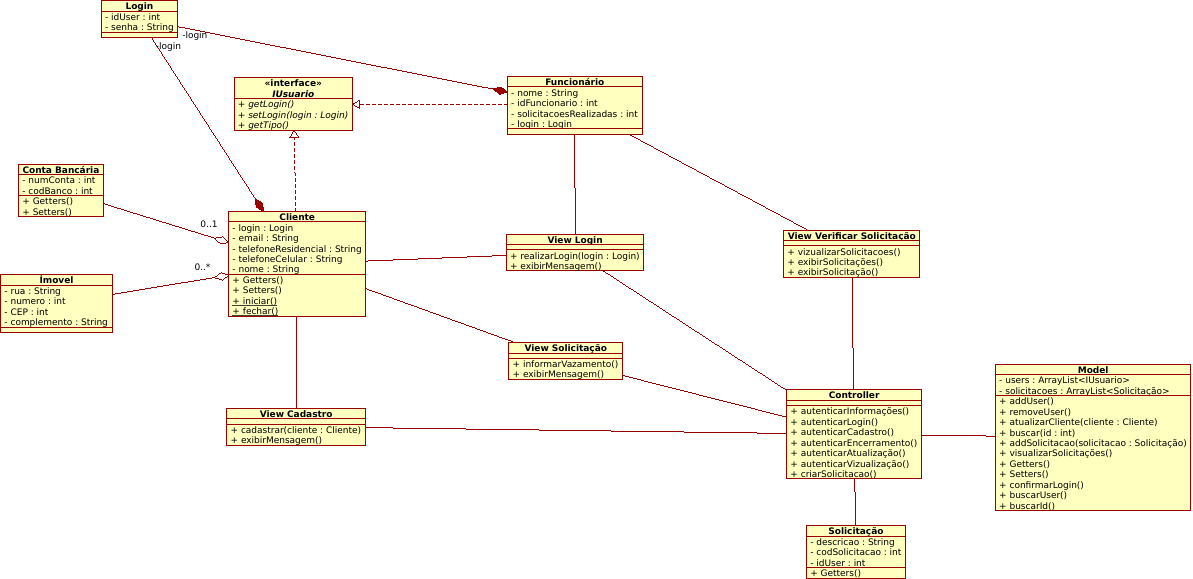


Figura - Diagrama de Classes

* Diagramas de Sequência
  + Diagrama

    Descrição gerada automaticamenteCadastrar

Figura . Diagrama de Sequência - Cadastrar

* + Diagrama

    Descrição gerada automaticamenteInformar Vazamento

Figura . Diagrama de Sequência - Informar Vazamento

* + Diagrama

    Descrição gerada automaticamenteRealizar Login

Figura . Diagrama de Sequência - Realizar Login

* + Diagrama

    Descrição gerada automaticamenteVerificar Solicitação

Figura . Diagrama de Sequência - Verificar Solicitação

* Diagrama de Estados
  + Diagrama

    Descrição gerada automaticamenteCadastro

Figura . Diagrama de Estado - Cadastro

* + Uma imagem contendo Gráfico

    Descrição gerada automaticamenteLogin
  + Diagrama

    Descrição gerada automaticamenteSolicitar Religamento

Figura . Diagrama de Estado - Solicitar Religamento

* + Solicitar Segunda via de conta

Diagrama

Descrição gerada automaticamente

Figura . Diagrama de Estados Solicitar Segunda Via de Conta

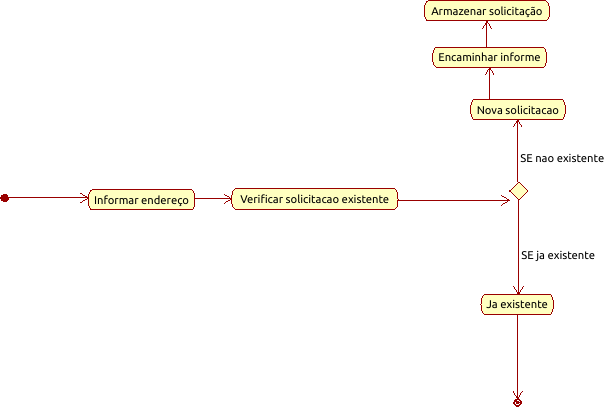
* Informar Vazamento

Figura . Diagrama de Estados - Informar Vazamento

* Solicitar Atendimento

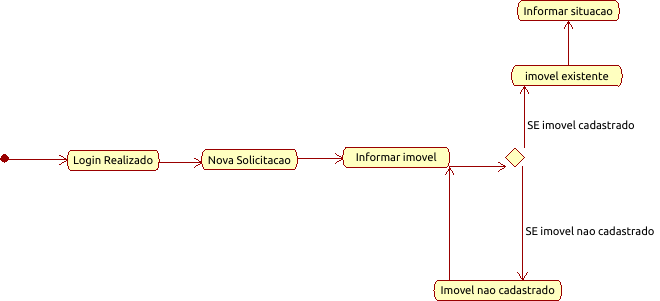


Figura . Diagrama de Estado - Solicitar Atendimento

1. **IMPLEMENTAÇÕES – POLIMORFISMO, DESIGN PATTERNS, CONTROLE DE ERROS E INTERFACE PARA INTEGRAÇÃO**

Para a implementação do código foi implementado, um design pattern, controle de erros e o polimorfismo, como solicitado pelo professor. O design pattern escolhido foi o **Singleton**, pois, após uma análise, ele foi essencial para a implementação do projeto. Ele foi implementado nas classes **Cliente** e **Controller**, fazendo com que durante todo o processo de execução do software o objeto das duas classes fossem o mesmo, exceto pela classe cliente, onde, caso o cliente escolhesse “sair” da sua conta, o objeto deixava de existir (voltando a assumir o valor **nulo**). Segue o trecho de código tanto da classe **Cliente** quanto da classe **Controller**.

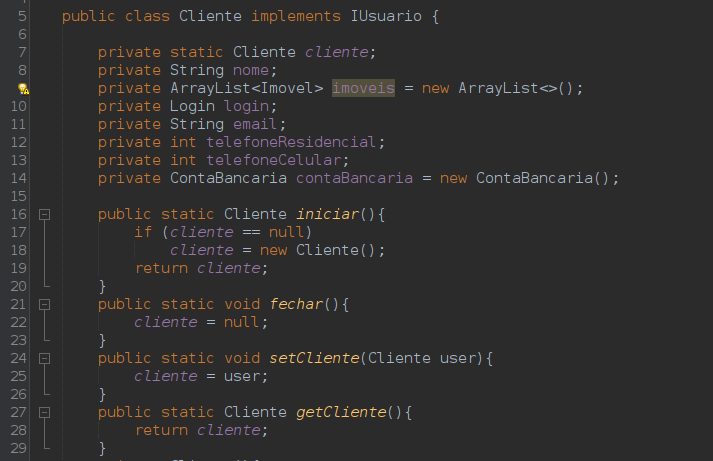


Figura . Singleton – Cliente

Texto

Descrição gerada automaticamente

Figura . Singleton – Controller

Vale ressaltar que estes são os principais usos de Singleton dentro do código do software, porém há outros casos de implementação de Singleton em janelas.

O controle de erros implementado foi para identificar letras em campos onde é necessário inserir somente números, prevenindo assim eventuais erros em tempo de execução, o método que implementa este controle de erros foi implementado na classe **Controller**, onde é recebido como parâmetro a **String** a ser “convertida” para inteiro, e caso ocorresse a exceção seria retornado o valor “-999”. Segue o trecho de código:

Texto

Descrição gerada automaticamente

Figura . Controle de Erros

O polimorfismo foi implementado na **Janela Principal,** onde, ao usuário realizar o login é detectado se ele é um **Cliente** ou **Funcionário**, dependendo do tipo, é aberto uma janela diferente para cada um. Segue o trecho de código da implementação:

Texto

Descrição gerada automaticamente

Figura . Polimorfismo

Após uma reunião com o grupo de integração, foi decidido acrescentar uma classe **Main** que implementa a interface passada por eles, sendo assim foi criado esta classe, em que todos os métodos pedidos são implementados, e em específico o método iniciar, é chamado a classe **Janela Principal** para inicializar o programa. Segue grande parte da classe **Main** para demostrar a implementação:

Texto

Descrição gerada automaticamente

Figura . Classe Main - Integração

Segue, o link para acesso ao projeto completo, via GitHub:

[filipeoioi/ProjetoFinalPOO: Projeto final da disciplina de Programação Orientada à Objeto, na qual é projetado um software de um Sistema de Gerenciamento de Água e Esgoto, para uma prefeitura. (github.com)](https://github.com/filipeoioi/ProjetoFinalPOO/tree/main)

1. **REFERÊNCIAS BIBLIOGRÁFICAS**

**G. Booch, J. Rumbaugh, I. Jacobson. UML, Guia do Usuário. Editora Campus, 2000.**

**H. M. Deitel, P. J. Deitel. Java: Como Programar, 8a. Edição. Pearson, 2010**