**FACULDADE INTERAMERICANA DE PORTO VELHO**

**PAULO JOSÉ DOS SANTOS**

**TRABALHO DE CONCLUSÃO DE CURSO**

**SISTEMA DE CONTROLE DE PACIENTES NO TRATAMENTO FORA DOMICÍLIO - SISPTD**

**Porto Velho**

**2016**

**PAULO JOSÉ DOS SANTOS**

**TRABALHO DE CONCLUSÃO DE CURSO**

**SISTEMA DE CONTROLE DE PACIENTES NO TRATAMENTO FORA DOMICÍLIO - SISPTD**

Trabalho de conclusão de curso, apresentado como requisito obrigatório para obtenção do grau de Bacharel na Faculdade Interamericana de Porto Velho – UNIRON no Curso de Sistemas de Informação.

**Orientadores: José Avani das Chagas Júnior, Hudyson Santos Barbosa e Vivaldo Pinto.**

**Porto Velho**

**2016**

**PAULO JOSÉ DOS SANTOS**

**SISTEMA DE CONTROLE DE PACIENTES NO TRATAMENTO FORA DOMICÍLIO - SISPTD**

**Trabalho de conclusão de curso apresentado a Faculdade Interamericana de Porto Velho – UNIRON como parte dos requisitos para obtenção do grau de bacharelado em Sistemas de Informação no curso de Sistemas de Informação.**

**­­­­**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Prof. Ms. Autran Dias Almeida**

**BANCA EXAMINADORA**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Prof. Esp. José Avani das Chagas.**

**Presidente da Banca e Orientador**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Prof. Esp. Hudyson Santos Barbosa.**

**Orientador**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Prof. Esp. Vivaldo Pinto. Título.**

**Orientador**

**DEDICATÓRIA**

Dedico esse trabalho a minha mãe Maria dos Santos Alves (em memoria), que sempre me orientou e aconselhou, pelo seu carinho estímulo que me ofereceu, dedico essa conquista como gratidão.

Para minha querida Esposa Edlaine Lima e meu Filho João Paulo Lima.

Às pessoas que direta e indiretamente influenciaram está conquista.

**AGRADECIMENTOS**

A Deus por tudo que me tem proporcionado todos os dias, e o qual me deu forças para chegar até aqui.

A minha amada Esposa pela paciência e compreensão pelas horas que me ausentei para dedicar a este trabalho.

Aos professores orientadores que me ajudaram compartilhando seus conhecimentos o qual foi fundamental para este trabalho.

**RESUMO**

O trabalho aqui apresentado visa documentar as etapas de desenvolvimento de software para controle de paciente no tratamento fora domicilio que será utilizado pela Secretaria de Estado de Saúde – SESAU. O sistema denomina-se SISPTD, a principal função do SISPTD é controlar o fluxo dos pacientes que dão inicio ao tratamento fora domicilio. Para a realização da analise do sistema foram utilizados as principais técnicas e ferramentas de desenvolvimento, tais como UML (Linguagem Unificada de Modelagem), c# (Csharp) como linguagem de programação, Microsoft Visual Studio 2013 com framework Asp.Net MVC como IDE(Ambiente de Desenvolvimento Integrado) e para a persistência de dados foi utilizado o Entity framework, para documentação e especificação da modelagem foi utilizado a ferramenta Enterprise Architect. Para a base de dados foi utilizado o SQL Server 2012. O padrão de arquitetura adotado para codificação do sistema foi o MVC que procura dividir as responsabilidades das camadas de codificação também foi utilizado uma camada denominada BO (Objeto de Negócio) Onde foi codificado as regras de negocio.

**Palavras-chave:** PACIENTE. MVC. UML. TRATAMENTO

**ABSTRACT**

The work presented here aims to document the software development steps for patient control treatment outside home, which will be used by the State Department of Health - SESAU. The system is called SISPTD, the main function of SISPTD is to control the flow of patients to give early treatment outside household. For the realization of the system analysis were used the main techniques and development tools such as UML (Unified Modeling Language) C # (C #) as the programming language, Microsoft Visual Studio 2013 with Asp.Net MVC framework as IDE (Environment Integrated Development) and the persistence of data we used the Entity framework for documentation and specification of the modeling was used to Enterprise Architect tool. For the database SQL Server was used 2012. The architectural pattern adopted for coding system was the MVC that seeks to divide the responsibilities of coding layers was also used a layer called BO (Business Object) Where was codified rules of business.

**Keywords:** PATIENT. MVC. UML. TREATMENT

**LISTA DE ILUSTRAÇÕES**

Figura 1 - Diagrama de Caso de Uso

Figura 2 - Diagrama de Atores

Figura 3 - Diagrama de Caso de Uso - Efetuar Login

Figura 4 - Diagrama de Caso de Uso - Manter Pessoa

Figura 5 - Diagrama de Caso de Uso - Consultar Pessoa

Figura 6 - Diagrama de Caso de Uso - Protocolar Documentação

Figura 7 - Diagrama de Caso de Uso - Manter Pericia

Figura 8 - Diagrama de Caso de Uso - Manter Agendamento

Figura 9 - Diagrama de Caso de Uso - Requisição de Viagem

Figura 10 - Diagrama de Caso de Uso - Manter Clinica

Figura 11 - Diagrama de Caso de Uso - Manter Médico

Figura 12 - Diagrama de Sequência - Efetuar Login

Figura 13 - Diagrama de Sequência - Manter Pessoa

Figura 14 - Diagrama de Sequência - Consultar Pessoa

Figura 15 - Diagrama de Sequência - Protocolar Documentação

Figura 16 - Diagrama de Sequência - Manter Pericia

Figura 17 - Diagrama de Sequência - Manter Agendamento

Figura 18 - Diagrama de Sequência - Requisição de Viagem

Figura 19 - Diagrama de Sequência - Manter Clinica

Figura 20 - Diagrama de Sequência - Manter Médico

SUMÁRIO

[1 INTRODUÇÃO 9](#_Toc447048700)

[2 ANÁLISE 10](#_Toc447048701)

[2.1 ENTREVISTAS 10](#_Toc447048702)

[2.2 DESCRIÇÃO DO SISTEMA EXISTENTE 10](#_Toc447048703)

[2.3 OBJETIVOS DO SISTEMA 11](#_Toc447048704)

[2.4 RESUMO EXECUTIVO 11](#_Toc447048705)

[2.5 REQUISITOS 11](#_Toc447048706)

[2.5.1 Requisitos funcionais 11](#_Toc447048707)

[2.5.2 Requisitos não funcionais 11](#_Toc447048708)

[2.5.3 Requisitos de Informação 11](#_Toc447048709)

[3 PROJETO 12](#_Toc447048710)

[3.1 CASOS DE USO 12](#_Toc447048711)

[3.1.1 Casos de uso principais 12](#_Toc447048712)

[3.1.2 Casos de uso secundários 12](#_Toc447048713)

[3.2 DIAGRAMAUML 12](#_Toc447048714)

[3.2.1 Diagramas de caso de uso 12](#_Toc447048715)

[3.2.2 Diagrama de sequência (cada caso de uso) 12](#_Toc447048716)

[3.2.3 Diagrama Conceitual 12](#_Toc447048717)

[3.2.4 Diagrama de Classe 12](#_Toc447048718)

[3.3 BANCO DE DADOS 12](#_Toc447048719)

[3.2.5 Diagrama Relacional 12](#_Toc447048720)

[4 IMPLEMENTAÇÃO 13](#_Toc447048721)

[4.1 CAMADA DE APRESENTAÇÃO 13](#_Toc447048722)

[4.2 CAMADA DE CODIFICAÇÃO DE CLASSES 13](#_Toc447048723)

[4.3 TESTE 13](#_Toc447048724)

[5 CONCLUSÃO 14](#_Toc447048725)

[REFERÊNCIAS 15](#_Toc447048726)

[APÊNDICE 16](#_Toc447048727)

# 1 INTRODUÇÃO

O Mecanismo de Tratamento Fora Domicílio, tem como objetivo encaminhar pacientes portadores de doenças não tratáveis no município de origem, sendo de responsabilidade da Gerencia de TFD todo o processo envolvido nesse Mecanismo, como exemplo, emitir uma requisição para a agência de viajem conveniada com Estado.

Após o paciente ter dado entrada com uma documentação necessária para o inicio do tratamento fora do seu domicílio, a Gerencia de TFD realiza uma pericia nos exames apresentados pelo paciente, só então é realizado o agendamento junto a clinica onde será realizado o tratamento e posteriormente a emissão da requisição de passagens. Para um resultado ainda mais eficaz, à necessidade de gerenciar o processo envolvido nesse mecanismo, com isso o surgimento do SISTPD – Sistema de Controle de Paciente no Tratamento Fora Domicílio.

Com o uso desse Sistema espera-se um processo mais ágil no Mecanismo de TFD, e evitar planilhas e anotações avulsas causando desencontro nas informações.

# 2 ANÁLISE

De acordo com Xexéo (2007, p. 16) entende-se que a análise é “a tarefa de levantar e descrever os requisitos de um sistema, definindo de que forma deve funcionar para atender as expectativas de todos que nele possuem algum interesse”.

Para Engholm Júnior (2010, p. 83) “procura-se entender o problema a partir da perspectiva do cliente, sem preocupações relacionadas à tecnologia que será utilizada ou ao design do sistema”.

O Sistema SISPTD resultado das necessidades apresentada pelo cliente com uma análise baseada em entrevistas com os interessados no sistema, onde foi possível fazer o levantamento dos requisitos do sistema.

# 2.1 ENTREVISTAS

As entrevistas servem para nos auxiliar no levantamento de requisitos. Bezerra (2007, p. 20) descreve: “o principal objetivo do levantamento de requisitos é que usuários e desenvolvedores tenham a mesma visão do problema a ser resolvido.”. De acordo Guedes (2011, p. 22) “devem ser realizadas tantas entrevistas quantas forem necessárias para que as necessidades do usuário sejam bem-compreendidas.”.

Para a elaboração do sistema SISPTD foram realizadas algumas entrevistas, a fim de obter informações para o desenvolvimento do software.

# 2.2 DESCRIÇÃO DO SISTEMA EXISTENTE

O Sistema atual conta com uma base de dados crítica e não confiável e ainda não sendo um sistema multiusuário, e isso tem tornado os processos envolvidos no TFD mais manual do que informatizado, e isso não tem passado confiança para os operadores do TFD que tem realizado inúmeras tarefas de modo arcaico.

# 2.3 OBJETIVOS DO SISTEMA

O Sistema deverá Controlar o processo de emissão de requisições de viagens dos Pacientes que darão Entrada no tratamento fora domicílio.

# 2.4 RESUMO EXECUTIVO

O sistema de controle de paciente no tratamento fora domicílio- SISPTD tem como seu objetivo principal controlar o processo que envolve o mecanismo de tratamento fora domicílio, e manter a informação pertinente aos pacientes que estão utilizando esse serviço. Quando o paciente der início ao tratamento será feito seu cadastro no sistema e caso esse mesmo paciente venha utilizar novamente o serviço de TFD o processo será mais rápido, pois o mesmo já terá suas informações no sistema, cada setor.

# 2.5 REQUISITOS

Sommerville (2007, p. 79) descreve: “os requisitos de um sistema são descrições dos serviços fornecidos pelo sistema e as suas restrições operacionais.”.

Para o SISPTD a Elicitação dos Requisitos foram feitas através de documentos e entrevistas que foram fornecidos pelo setor de TFD.

# 2.5.1 Requisitos funcionais

Xexéo (2007, p. 44) descreve: “Um requisito funcional representa algo que o sistema deve fazer, ou seja, uma função esperada do sistema que agregue algum valor a seus usuários.”. Sommerville (2007, p. 80) explica que: “Os requisitos funcionais são declarações de serviços que o sistema deve fornecer, como o sistema deve reagir as entradas especificas e como o sistema deve se comportar em determinadas situações.”

Os requisitos funcionais identificados no SISPTD são:

* Efetuar Login
* Manter Pessoa
* Manter Médico
* Manter Funcionário
* Manter Processo
* Manter Pericia
* Manter Clinica
* Manter Agendamento
* Requisição de Viagem

# 2.5.2 Requisitos não funcionais

Para Xexéo (2007, p. 44) “requisitos não funcionais falam da forma como os requisitos funcionais devem ser alcançados.”. Já para Bezerra (2007, p. 21) “Requisitos não funcionais declaram as características de qualidade que o sistema deve possuir e que estão relacionadas às suas funcionalidades.”.

Os requisitos não funcionais do SISPTD estão especificados abaixo:

**Hardware:**

Impressoras laser com placa de rede para impressão de ficha de atendimento e requisição de Passagens.

Para a instalação do servidor Web no local será necessário um computador com no mínimo 8GB de RAM, Processador I5 ou superior.

Para a instalação do servidor de Banco de Dados no local será necessário um computador com no mínimo 8GB de RAM, Processador I5 ou superior.

**Software**:

Windows Server 2012 ou superior para servir como servidor de aplicação;

Windows Server 2012 com SQL Server 2012 ou superior para servir como servidor de Banco de Dados;

# 2.5.3 Requisitos de Informação

Para Xexéo (2007, p.44) “Requisitos de Informação devem representa a informação que o cliente deseja obter do sistema”, os requisitos de informação do SISPTD procura atender no controle de requisições emitidas para pacientes que precisam realizar procedimentos médicos fora de sua Unidade Federativa.

# 3 PROJETO

Para Bezerra (2007, p. 25) “Na fase projeto é que se determina como o sistema funcionará para atender aos requisitos, de acordo com os recursos tecnológicos existentes”. Ainda para Bezerra é na fase de projeto que deve ser considerado os aspectos físicos e dependentes de implementação. No projeto do Sistema de Controle de Paciente no Tratamento Fora Domicilio (SISPTD), foram feitas visitas in-loco, entrevistas com os envolvidos da gerencia de Tratamento fora Domicilio, também foram analisado documentos do processo de tratamento fora domicilio para então ser elaborado o projeto do sistema.

# 3.1 CASOS DE USO

Para Guedes (2011, p. 52) “O diagrama de caso de uso procura, por meio de uma linguagem simples, possibilitar a compreensão do comportamento externo do sistema”, já Bezerra (2007, p. 46) explica que: “caso de uso é a especificação de uma sequência de interações entre um sistema e os agentes externos que utilizam esse sistema”. Os casos de uso do SISPTD foram elaborados conforme os requisitos levantados nas entrevistas e estão descrito abaixo.

# 3.1.1 Casos de uso principais

Caso de uso principal ou primário é considerado o mais importante, por que tem a finalidade representar o fluxo principal da empresa que está sendo automatizada. De acordo com Guedes (2011, p. 54) “Um caso de uso é considerado principal, conhecido também como caso de uso primário quando se refere a um processo importante, que enfoca um dos requisitos funcionais do software”. Já para Bezerra (2007, p. 61) “Casos de uso primários são aqueles que representam os objetivos dos atores”.

**Quadro 1** – Caso de uso Requisição de Passagem

|  |  |  |
| --- | --- | --- |
| **Nome** | Requisição de Passagem | N-01 |
| **Sumário** | Descreve as interações para emissão de requisição de viajem | |
| Ator primário: Funcionário  Ator secundário: Gerente  Pré-Condição:   1. estar devidamente logado no sistema 2. Paciente deve estar devidamente agendado   Pós-Condição: Uma requisição é registrada no sistema | | |
| **Fluxo Principal** | | |
| 1. O Ator seleciona a opção requisição;  2. O sistema exibe o formulário para o preenchimento das informações da requisição;  3. O Ator informa o CPF do Paciente para realizar uma busca;  4. O Sistema retorna as informações do Paciente do CPF que foi informado;  5. O Ator Seleciona a opção adicionar acompanhante;  6. O Sistema exibe a tela de adição de acompanhante;  7. O Ator informa o CPF do Acompanhante  8. O Sistema retorna as informações do Acompanhante;  9. O Ator Seleciona o botão adicionar  10. O Sistema Adiciona o Acompanhante na Lista de Acompanhantes da requisição;  11. O Ator Seleciona a opção Gravar;  12. O Sistema valida os campos obrigatórios;  13. O Sistema retorna uma mensagem de sucesso.  13. O Caso de Uso é finalizado. | | |
| **Fluxo Alternativo** | | |
| 1. O Sistema informa através de mensagem que o paciente não possui agendamento;  2. O Sistema exibe a opções para o Ator manter o Agendamento do Paciente;  3. O Caso de Uso é finalizado. | | |
| **Fluxo de Exceção** | | |
| [Passo 12] Validação dos campos obrigatórios;  12. O Sistema Valida que campos obrigatórios não foram preenchidos;  12.1 O Sistema emite uma mensagem informando sobre os campos obrigatórios e não efetua o cadastro da requisição. | | |

Fonte: O Autor.

# 3.1.2 Casos de uso secundários

Para Guedes (2011, p. 54) “Caso de uso secundário se refere a um processo periférico”, porém não menos importante do que o caso de uso principal, já (Bezerra 2007, p.63)” O caso de uso secundário é aquele que não traz benefício direto para os atores, mas que é necessário para o funcionamento do sistema”, no SISPTD foram identificado os casos de uso secundário listados abaixo.

**Quadro 2** – Caso de uso Efetuar Login

|  |  |  |
| --- | --- | --- |
| **Nome** | Efetuar Login | N-02 |
| **Sumário** | O Ator deverá Informar seu login e senha que foi previamente cadastrado pelo Gerente. | |
| Ator primário: Funcionário  Ator secundário: Gerente  Pré-Condição:   1. Ator deve informar um nome de login e uma senha.   Pós-Condição:  O Ator devidamente autenticado no sistema. | | |
| **Fluxo Principal** | | |
| 1. O Sistema apresenta um formulário contendo os campos login e senha;  2. O Ator informa seu login e senha e seleciona o botão entrar;  3. O Sistema Valida as informações dos campos do formulário;  4. O Sistema apresenta sua tela principal;  5. O Caso de Uso é finalizado; | | |
| **Fluxo Alternativo** | | |
| 1. O Sistema informa através de mensagem, login ou senha inválida.  2. O Sistema exibe a opções para o Ator recuperar senha;  3. O Ator seleciona a opção para recuperar senha;  4. O Sistema exibe o formulário com as informações necessárias para recuperar a senha;  5. O Ator preenche as informações solicitadas no formulário e efetua o registro;  6. O Sistema Valida as informações do formulário e retorna a mensagem “Sucesso!”;  7. O Caso de Uso é finalizado. | | |
| **Fluxo de Exceção** | | |
| [Passo 3] Validação das Informações;  3. O Sistema verifica se o Login e a senha informada correspondem aos mesmos que foram cadastrados; | | |

Fonte: O Autor.

**Quadro 3** – Caso de uso Manter Paciente

|  |  |  |
| --- | --- | --- |
| **Nome** | Manter Paciente | N-03 |
| **Sumário** | Neste caso de uso o Ator irá Manter as Informações de um paciente no sistema. | |
| Ator primário: Funcionário  Ator secundário: Gerente  Pré-Condição:   1. O Ator deve estar devidamente autenticado no sistema, e ter permissão para executar o caso de uso manter Paciente.   Pós-Condição:  O Paciente será mantida no sistema. | | |
| **Fluxo Principal** | | |
| 1. O Ator seleciona a opção de cadastrar paciente;  2. O Sistema exibe o formulário com as informações para ser mantido um paciente no sistema  3. O Ator preenche as informações solicitadas no formulário e efetua o registro;  4. O Sistema valida os campos obrigatórios do formulário;  5. O Sistema retorna uma mensagem de “Registrado com Sucesso!”;  6. O Caso de Uso é finalizado. | | |
| **Fluxo Alternativo** | | |
| 1. O sistema indica e retorna através de mensagem que existem campos inválidos;  2. O Ator insere ou altera as informações dos campos errados que foram indicados pelo sistema e efetua o registro.  3. O Sistema valida os campos dos formulários e retorna uma mensagem “Registrado com sucesso!”;  4. O Caso de Uso é finalizado. | | |
| **Fluxo de Exceção** | | |
| Continua | | |
| [Passo 4] Validação dos campos obrigatórios;  4. O Sistema Valida que campos obrigatórios não foram preenchidos;  4.1. Verifica se o CPF informado é valido;  4.2. Verifica se pelo menos um dos campos foi preenchido  4.3. Verifica se o campo cartão SUS está devidamente preenchido.  4.4. Verifica se já não existe o CPF informado cadastrado.  4.5. O Sistema emite uma mensagem informando sobre os campos obrigatórios e não efetua o cadastro do paciente. | | |

Fonte: O Autor.

**Quadro 4** – Caso de uso Manter Pericia

|  |  |  |
| --- | --- | --- |
| **Nome** | Manter pericia | N-04 |
| **Sumário** | Neste caso de uso o Ator irá Manter a Informações referentes a perícia realizada no processo do paciente. | |
| Ator primário: Médico  Ator secundário: Gerente  Pré-Condição:   1. O Ator deve estar devidamente autenticado no sistema, e ter permissão para executar o caso de uso manter perícia.   Pós-Condição:  Será mantido as informações da perícia do Paciente. | | |
| **Fluxo Principal** | | |
| 1. O Ator seleciona no menu a opção de cadastrar perícia;  2. O Sistema exibe o formulário com as informações para ser mantido a perícia no sistema  3. O Ator preenche as informações solicitadas no formulário e efetua o registro;  4. O Sistema valida os campos obrigatórios do formulário;  5. O Sistema retorna uma mensagem de “Registrado com Sucesso! ”;  6. O Caso de Uso é finalizado. | | |
| **Fluxo Alternativo** | | |
|  | | |
| **Fluxo de Exceção** | | |
| Continua | | |
| [Passo 4] Validação dos campos obrigatórios;  4. O Sistema Valida que campos obrigatórios não foram preenchidos;  4.1. Verifica se o CPF e nome do paciente informado é valido;  4.2. Verifica se os campos com os dados do médico foram preenchidos  4.3. Verifica se o campo cartão SUS está devidamente preenchido.  4.5. O Sistema emite uma mensagem informando sobre os campos obrigatórios e não efetua o cadastro da perícia. | | |

Fonte: O autor.

**Quadro 5** – Caso de uso Manter Agendamento

|  |  |  |
| --- | --- | --- |
| **Nome** | Manter Agendamento | N-05 |
| **Sumário** | Este caso de uso tem como objetivo manter os agendamentos dos pacientes que irão usar o serviço de Tratamento Fora Domiciliar. | |
| Ator primário: Funcionário  Ator secundário: Gerente  Pré-Condição:   1. O Ator deve estar devidamente autenticado no sistema, e ter permissão para executar o caso de uso manter agendamento.   Pós-Condição:  Será mantido as informações do agendamento do Paciente. | | |
| **Fluxo Principal** | | |
| 1. O Ator seleciona o menu agendamento e seleciona o botão novo agendamento;  2. O Sistema exibe o formulário com as informações para ser mantido o agendamento no sistema;  3. O Ator preenche as informações solicitadas no formulário e efetua o registro;  4. O Sistema valida os campos obrigatórios do formulário;  5. O Sistema retorna uma mensagem de “Registrado com Sucesso! ”;  6. O Caso de Uso é finalizado. | | |
| **Fluxo Alternativo** | | |
|  | | |
| **Fluxo de Exceção** | | |
| Continua | | |
| [Passo 4] Validação dos campos obrigatórios;  4. O Sistema Valida que campos obrigatórios não foram preenchidos;  4.1. Verifica se a data fornecida é uma data valida;  4.2. Verifica se os campos com os dados do paciente foram preenchidos  4.3. O Sistema exibe uma mensagem informando sobre os campos obrigatórios e não efetua o cadastro da perícia. | | |

Fonte: O autor.

**Quadro 6** – Caso de uso Manter Clínica

|  |  |  |
| --- | --- | --- |
| **Nome** | Manter Clínica | N-06 |
| **Sumário** | Este caso de uso tem como objetivo manter Informações sobre as clinicas e hospitais onde os pacientes irão usar o serviço de Tratamento Fora Domiciliar. | |
| Ator primário: Funcionário  Ator secundário: Gerente  Pré-Condição:   1. O Ator deve estar devidamente autenticado no sistema, e ter permissão para executar o caso de uso manter clínica.   Pós-Condição:  Será mantido as informações da clínica no sistema. | | |
| **Fluxo Principal** | | |
| 1. O Ator seleciona o menu clínica/hospital e seleciona o botão cadastrar;  2. O Sistema exibe o formulário com as informações para ser mantido a clínica no sistema;  3. O Ator preenche as informações solicitadas no formulário e efetua o registro;  4. O Sistema valida os campos obrigatórios do formulário;  5. O Sistema retorna uma mensagem de “Registrado com Sucesso! ”;  6. O Caso de Uso é finalizado. | | |
| **Fluxo Alternativo** | | |
|  | | |
| **Fluxo de Exceção** | | |
| [Passo 4] Validação dos campos obrigatórios;  4. O Sistema Valida que campos obrigatórios não foram preenchidos;  4.1. Verifica se o campo cidade foi preenchido corretamente;  4.2. Verifica se o campo telefone está preenchido corretamente;  4.3. O Sistema exibe uma mensagem informando sobre os campos obrigatórios e não efetua o cadastro da perícia. | | |

Fonte: O autor.

**Quadro 7** – Caso de uso Manter Médico

|  |  |  |
| --- | --- | --- |
| **Nome** | Manter Médico | N-07 |
| **Sumário** | Este caso de uso tem por finalidade manter informações dos médicos que prestam serviço para Gerencia de Tratamento Fora Domiciliar. | |
| Ator primário: Gerente  Ator secundário:  Pré-Condição:   1. O Ator deve estar devidamente autenticado no sistema, e ter permissão para executar o caso de uso manter Médico.   Pós-Condição:  Será mantido as informações da clínica no sistema. | | |
| **Fluxo Principal** | | |
| 1. O Ator seleciona o menu medico e seleciona o botão cadastrar;  2. O Sistema exibe o formulário com as informações para ser mantido o médico no sistema;  3. O Ator preenche as informações solicitadas no formulário e efetua o registro;  4. O Sistema valida os campos obrigatórios do formulário;  5. O Sistema retorna uma mensagem de “Registrado com Sucesso! ”;  6. O Caso de Uso é finalizado. | | |
| **Fluxo Alternativo** | | |
|  | | |
| **Fluxo de Exceção** | | |
| [Passo 4] Validação dos campos obrigatórios;  4. O Sistema Valida que campos obrigatórios não foram preenchidos;  4.1. Verifica se o CPF informado é valido;  4.2. Verifica se um dos campos telefone está preenchido corretamente.  4.3. Verifica se o campo cartão SUS está devidamente preenchido.  4.4. Verifica se já não existe o CPF informado cadastrado.  4.5. Verifica se o campo crm está preenchido corretamente.  4.6. O Sistema emite uma mensagem informando sobre os campos obrigatórios e não efetua o cadastro do paciente. | | |

Fonte: O autor.

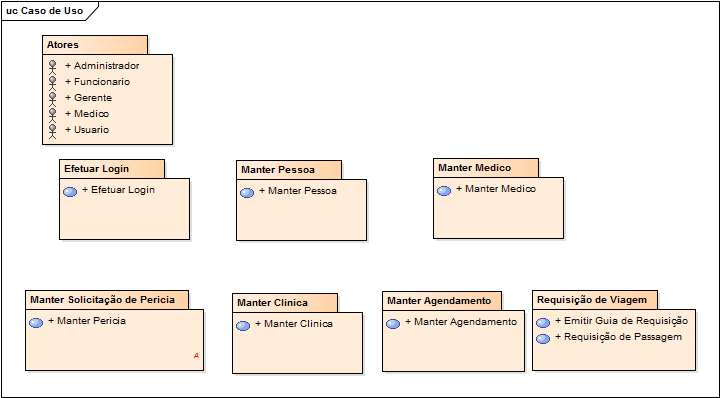
# 3.2 DIAGRAMA UML

Para Guedes (2011, p.19) “A UML é uma linguagem visual utilizada para modelar softwares baseados no paradigma de orientação a objetos.” Para um entendimento claro do software desenvolvido o SISPTD, foi utilizado em sua modelagem alguns dos diagramas da UML. Segundo (Bezerra 2007, p.14) “Pode-se fazer uma analogia da UML com uma caixa de ferramenta. Um construtor usa sua caixa de ferramentas para realizar suas tarefas.”

# 3.2.1 Diagramas de caso de uso

Para Bezerra (2007, p. 45) “Um Caso de uso é a especificação de uma sequência de interações entre um sistema e os agentes externos que utilizam esse sistema.”. Já para Guedes (2011, p. 52) “Esse diagrama tem por objetivo apresentar uma visão externa geral das funcionalidades que o sistema deverá oferecer aos usuários, sem se preocupar com a questão de como tais funcionalidades serão implementadas.”.

**Figura 1** – Diagrama de Caso de Uso

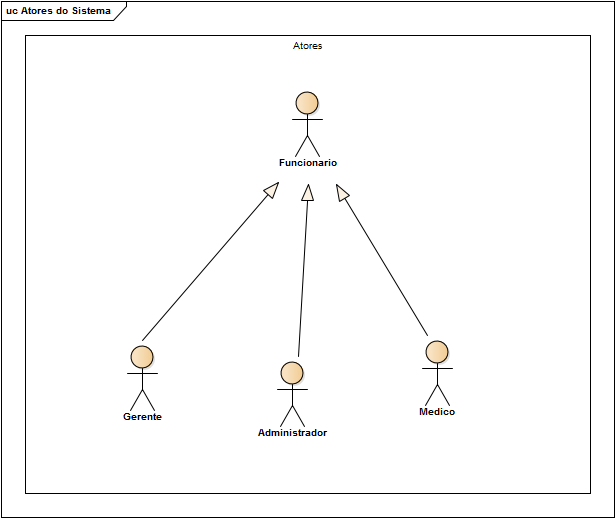


Fonte: O autor

**Atores**

Abaixo a figura 2 mostra o diagrama de atores que representa todos os atores responsáveis pela interação com o sistema.

**Figura 2** – Diagrama de Atores



Fonte: O autor

*Administrador*:

Ator responsável por pela administração de funções de acesso ao sistema.

*Funcionário:*

Esse Ator é responsável por manter os pacientes no sistema.

*Gerente:*

Ator Responsável por manter os Usuários do Sistema.

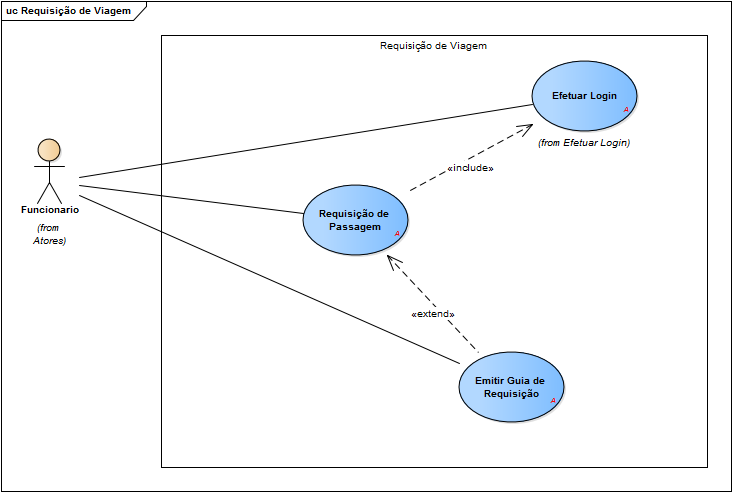
*Medico:*

Ator responsável pelas pericias dos pacientes.

**Efetuar Requisição de Passagem**

A figura3 mostra o diagrama de caso de uso efetuar requisição de passagem, que ilustra o ator funcionário emitindo a requisição de passagem no sistema.

**Figura 3** - Caso de Uso Efetuar requisição de passagem

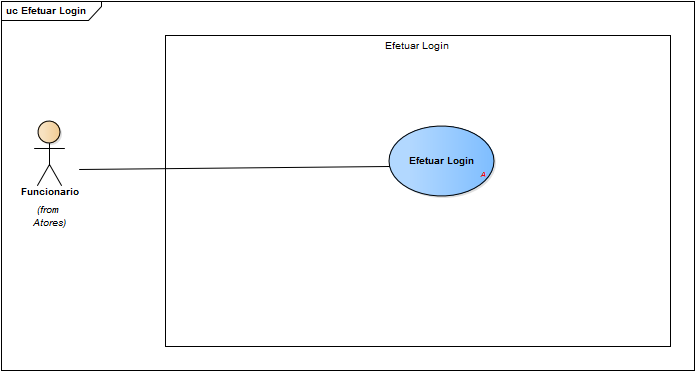


Fonte: O autor

**Efetuar Login**

A figura4 mostra o diagrama de caso de uso efetuar Login, que ilustra o ator funcionário autenticando-se no sistema.

**Figura 4** - Diagrama de caso de uso Efetuar Login

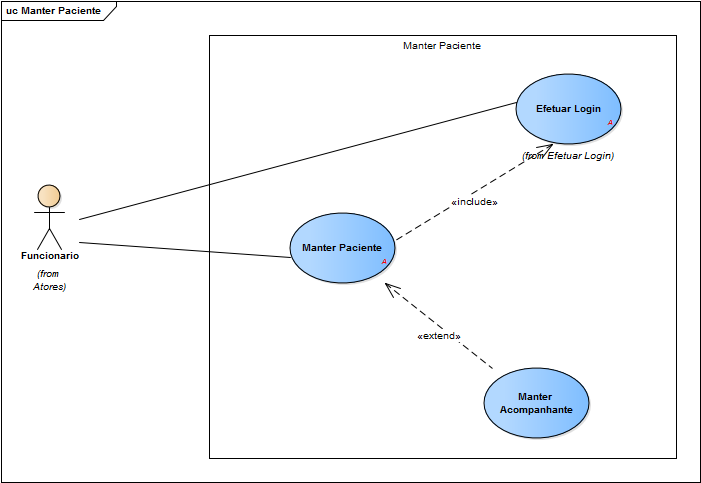
****

Fonte: O autor

**Manter Paciente**

Diagrama de caso de uso manter paciente mostrado na figura5, ilustra o procedimento para manter um paciente no sistema.

**Figura 5** - Diagrama de Caso de Uso Manter Paciente

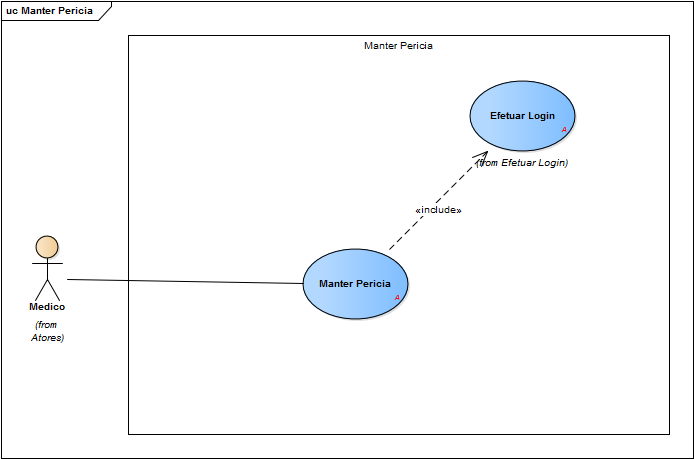


Fonte: O ator

**Manter Pericia**

Este diagrama de caso de uso ilustrado na figura6, tem por finalidade manter as informações obtidas pela perícia feita pelos médicos do GTFD.

**Figura 6** - Diagrama de Caso de Uso Manter Perícia

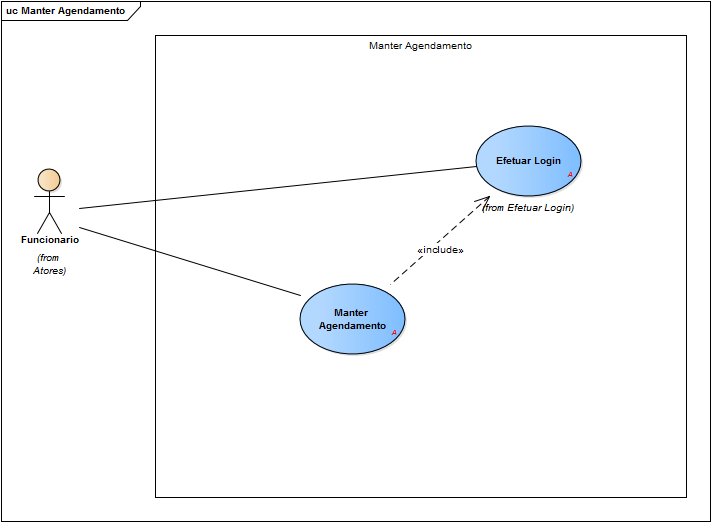


Fonte: O autor

**Manter Agendamento**

Este diagrama de caso de uso tem como objetivo manter os agendamentos dos pacientes que irão usar o serviço de Tratamento Fora Domiciliar.

**Figura 7** - Diagrama de Caso de Uso Manter Agendamento

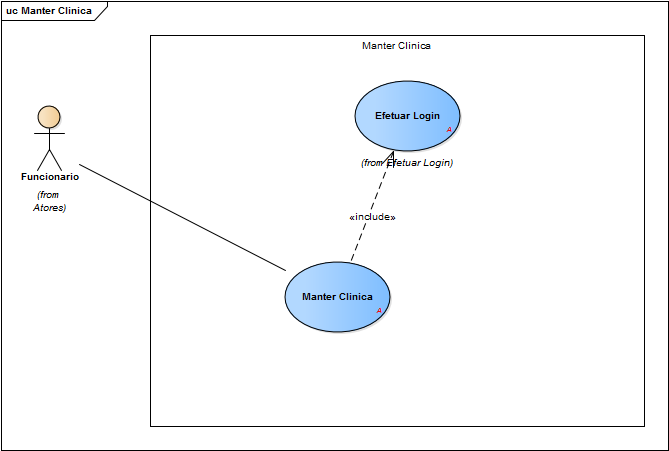


Fonte: O autor

**Manter Clínica**

O diagrama de caso de uso demostrado na figura8 tem como objetivo ilustrar como é mantido as clinicas no sistema.

**Figura 8** - Diagrama de Caso de Uso manter clínica

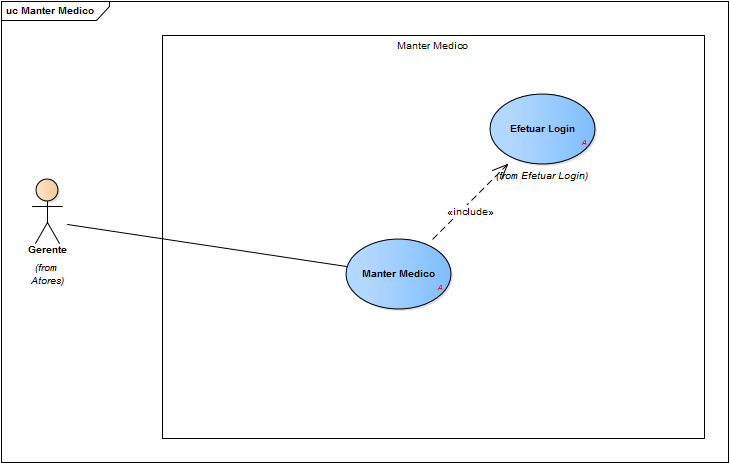


Fonte: O autor.

**Manter Médico**

O diagrama de caso de uso manter médico ilustrado na figura 9 descreve o processo para manter as informações do médico no sistema.

**Figura 9** - Diagrama de caso de uso manter Médico

****

Fonte: O autor.

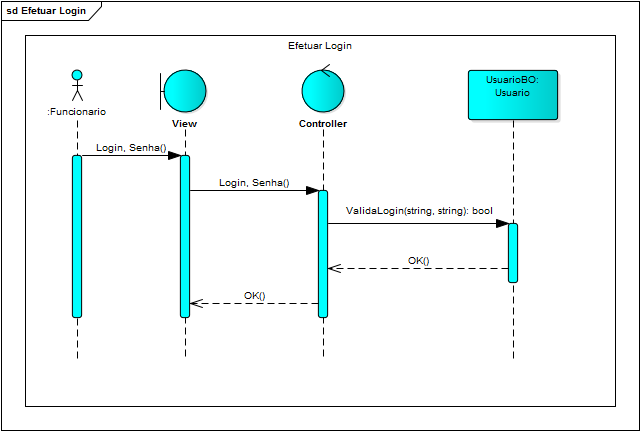
# 3.2.2 Diagrama de sequência

Segundo Guedes (2009, p.192) “Esse é um diagrama comportamental da UML que procura determinar a sequência de eventos que ocorre em determinada parte do software, identificando quais mensagens devem ser disparadas entre os elementos envolvidos e em que ordem.”

3.2.2.1 Diagrama de sequência caso de uso efetuar Login

A figura 9 exibe o diagrama de sequência efetuar Login, onde somente os funcionários que teve um cadastro prévio pode ter acesso ao sistema.

**Figura 10 -** Diagrama de Sequencia efetuar Login.

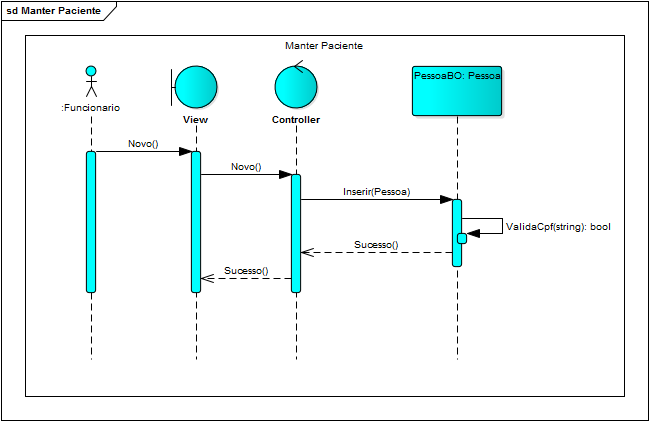


Fonte: O autor.

3.2.2.2 Diagrama de sequência caso de uso manter Paciente

A figura10 mostra o diagrama de sequência, onde exibe o fluxo de interação do usuário funcionário com o sistema, onde estar sendo mantido um paciente no sistema.

**Figura 11** - Diagrama de sequência manter Paciente

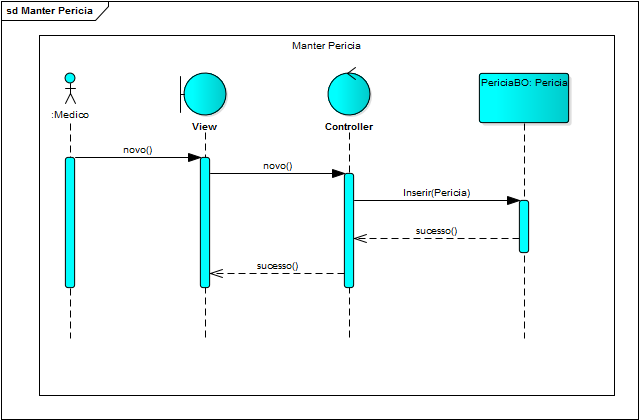


Fonte: O autor.

**3.2.2.3 Diagrama de sequência caso de uso manter Pericia**

A figura11 exibe o diagrama de sequência manter pericia onde deve ser mantido as informações das pericias realizadas nos pacientes.

**Figura 12** - Diagrama de sequência manter Pericia

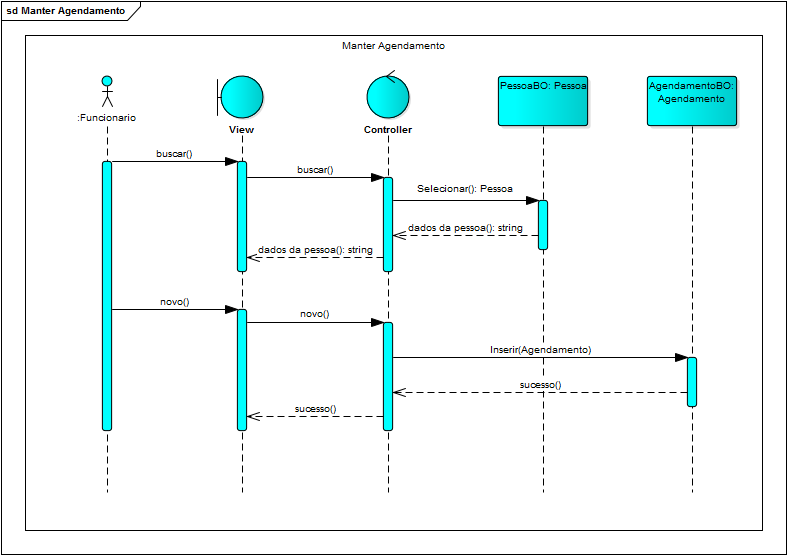


Fonte: O autor.

**3.2.2.4 Diagrama de sequência caso de uso manter Agendamento**

A figura12 demostra através do diagrama de sequência manter agendamento uma interação do usuário funcionário com o sistema onde está sendo mantido as informações do agendamento do paciente.

**Figura 13** - Diagrama de sequência manter Agendamento

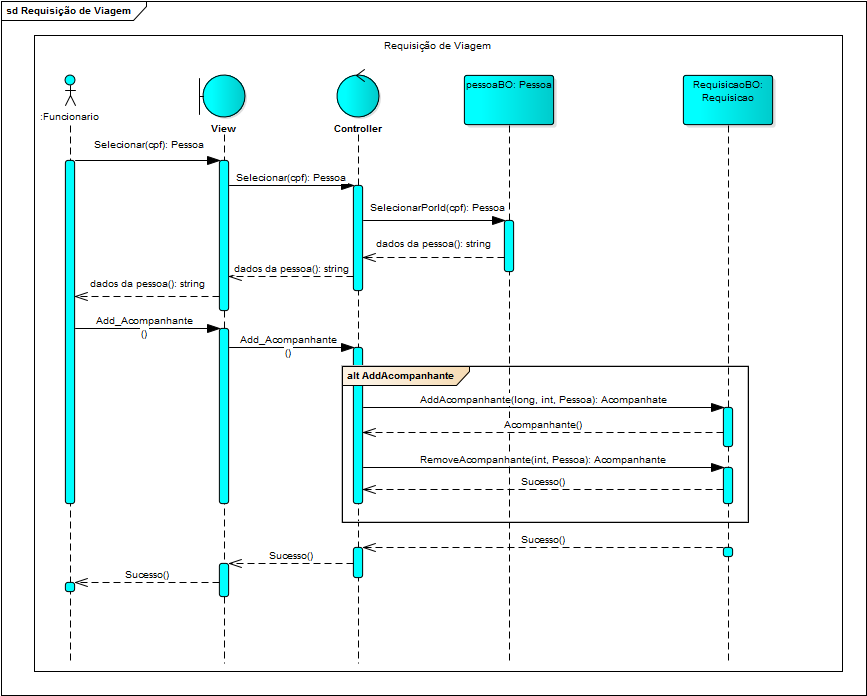


Fonte: O autor.

**3.2.2.5 Diagrama de sequência caso de uso requisição de Viagem**

A figura13 exibe o diagrama de sequência requisição de Viagem, onde descreve o processo de emissão de requisição de viagem.

**Figura 14** - Diagrama de sequência requisição de Viagem

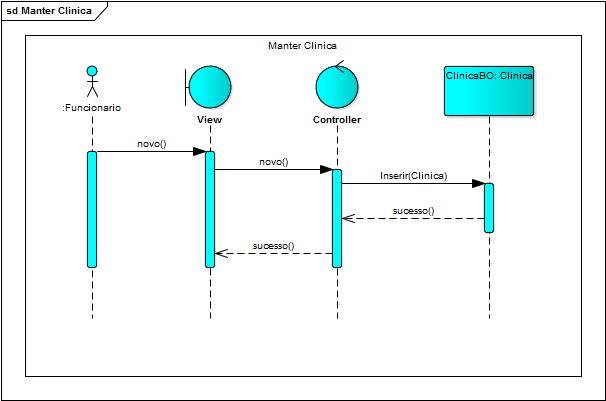


Fonte: O autor.

**3.2.2.6 Diagrama de sequência caso de uso manter Clinica**

A figura14 exibe o diagrama de sequência manter clínica que descreve o processo de manter as informações de uma clínica no sistema.

**Figura 15** - Diagrama de sequência manter Clinica

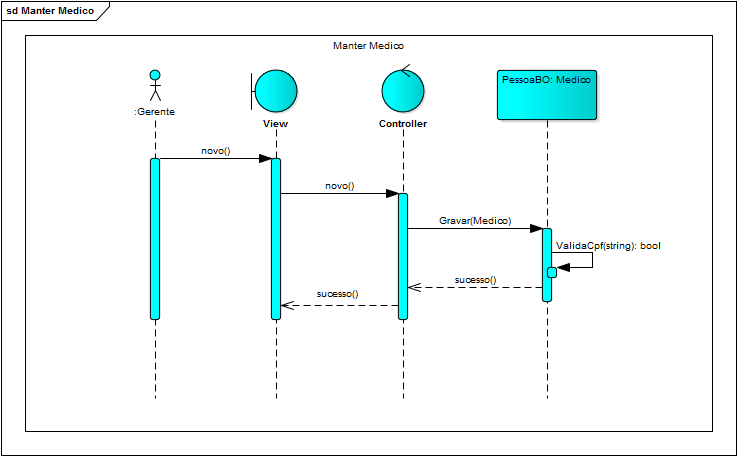


Fonte: O autor.

**3.2.2.7 Diagrama de sequência caso de uso manter Médico**

A figura 16 exibe o diagrama de sequência manter Médico que descreve o processo onde se mantem as informações do Médico.

**Figura 16** - Diagrama de sequência manter Médico

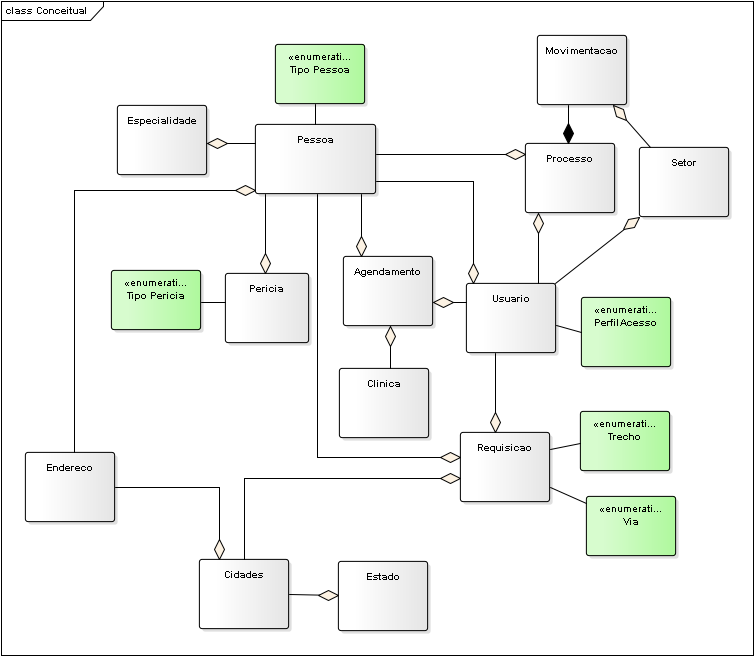


Fonte: O autor.

# 3.2.3 Diagrama Conceitual

O diagrama conceitual procura mostrar por meio de uma estrutura estática como estarão relacionadas às classes do sistema. Para Wazlawick (2011, p. 90) “O Modelo conceitual deve descrever a informação que o sistema vai gerencia. Tratando de um artefato do domínio do problema e da solução.”.

Fonte: O autor.

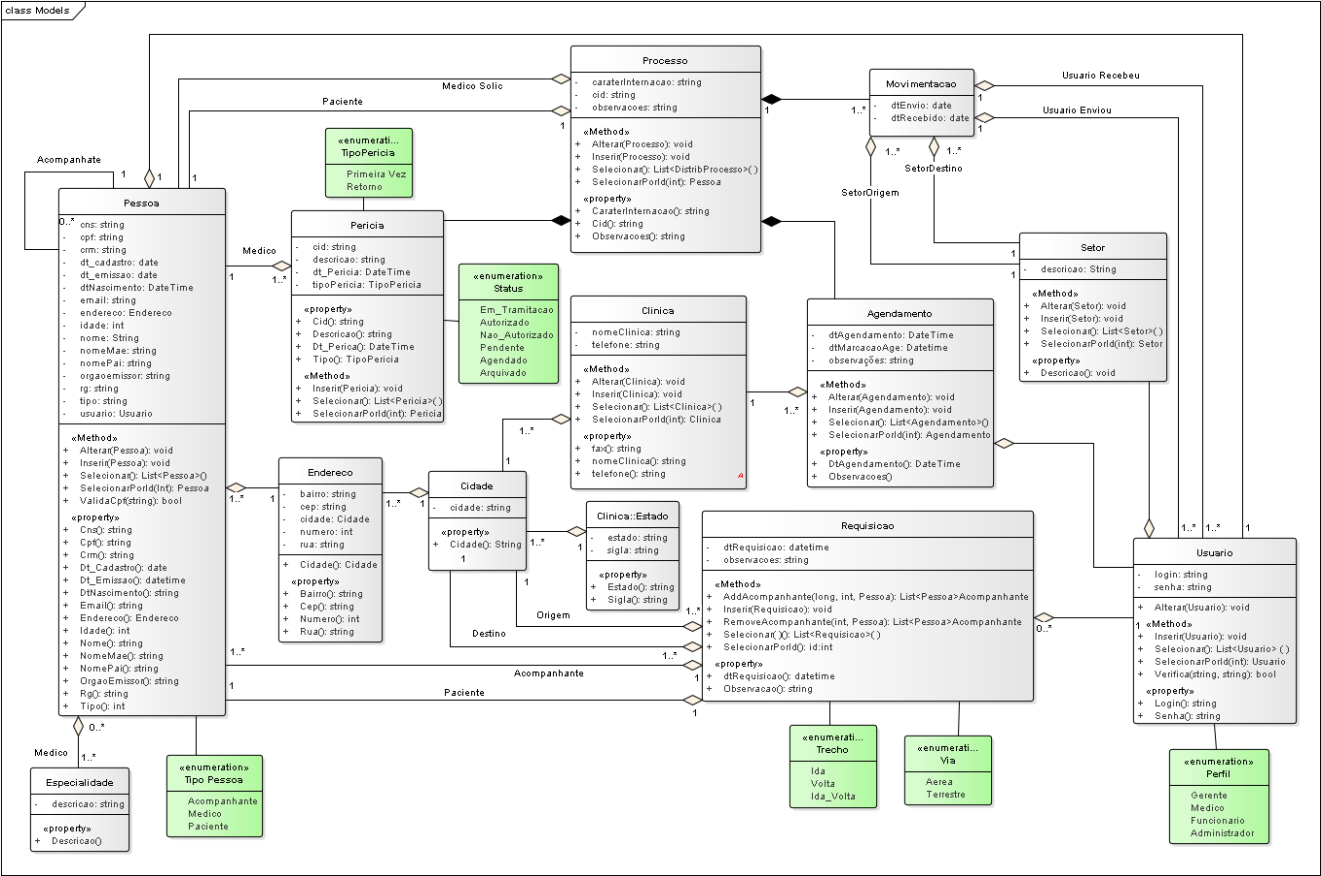


**Figura 17** - Diagrama Conceitual

# 3.2.4 Diagrama de Classe

Com o objetivo de demostrar graficamente as classes e suas funcionalidades servira como base para constituir as classes do sistema. Afirma Guedes (2011, p. 101) “O principal enfoque do diagrama de classe é permitir a visualização das classes que comporão o sistema com seus respectivos atributos e métodos, bem como em demonstrar como as classes do diagrama se relacionam, complementam e transmitem informações entre si.”.

**Figura 18** - Diagrama de Classe

Fonte: O autor.

**Classe Pessoa**

O quadro 8 mostra os atributos da classe Pessoa.

**Quadro 8** – Atributos da classe Pessoa

|  |  |  |
| --- | --- | --- |
| **Atributo** | **Tipo** | **Nota** |
| pessoaId | long | Atributo responsável por identificação única da pessoa. |
| pessoaPai | long | Atributo responsável pela identificação acompanhante do paciente. |
| Tipo | int | Atributo responsável por identificar o tipo de pessoa. |
| dt\_Cadastro | date | Atributo responsável por receber a data do cadastro. |
| Cpf | string | Atributo responsável por identificar o cpf da pessoa |
| Crm | string | Atributo responsável por identificar o crm, quando a pessoa for do tipo Médico. |
| Cns | string | Atributo responsável por identificar o cartão SUS da pessoa. |
| Rg | string | Atributo responsável por identificar o rg da pessoa. |
| Orgaoemissor | string | Atributo responsável por especificar o órgão emissor do rg cadastrado. |
| Dt\_Emissao | date | Atributo responsável por receber a data de emissão do rg. |
| Nome | string | Atributo responsável por identificar o nome da pessoa. |
| Dt\_Nascimento | date | Atributo responsável por identificar a data de nascimento da pessoa. |
| Idade | int | Atributo responsável por receber o resultado do cálculo da idade da pessoa. |
| Email | string | Atributo que recebe o e-mail da pessoa. |
| Nome\_Pai | string | Atributo responsável por receber o nome do pai da pessoa. |
| Nome\_Mae | string | Atributo responsável por identificar o nome da mãe da pessoa. |
| Tel | string | Atributo responsável por receber o telefone da pessoa |
| Cel | string | Atributo que especifica o celular cadastrado para pessoa. |
|  |  |  |

Fonte: O ator.

**Classe Distribuição de Processo**

Quadro 9 demostra os atributos da classe Distribuição de Processo

**Quadro 9** – Atributos da classe Distribuição de Processo

|  |  |  |
| --- | --- | --- |
| **Atributo** | **Tipo** | **Nota** |
| Distrib\_processoId | long | Atributo responsável pela identificação única da classe distribuição de processos. |
| SetorOrigem | long | Atributo responsável por receber o valor que determina o setor de origem do processo. |
| SetorDestino | long | Atributo que recebe o por receber o valor do id do setor de destino do processo. |
| Observacoes | string | Atributo que recebe um texto, caso o tramite do processo necessite de alguma justificativa. |
| PessoaId | long | Atributo que responsável por especificar o id da pessoa que identifica o paciente envolvido no processo. |
| UsuarioEnviou | long | Atributo que especifica o usuário do sistema que enviou o processo em andamento. |
| UsuarioRecebeu | long | Atributo que vai receber o id do usuário do sistema que recebeu o processo em andamento. |

Fonte: O autor.

**Classe Pericia**

O quadro 10 mostra os atributos da classe Pericia.

**Quadro 10** – Atributos da classe Pericia

|  |  |  |
| --- | --- | --- |
| **Atributo** | **Tipo** | **Nota** |
| PericiaId | long | Atributo que garante a unicidade da classe Pericia |
| Cid | string | Atributo que vai receber o cid que especifica as doenças que será descriminada na perícia. |
| Dt\_Pericia | date | Atributo que especifica a data em que foi realizado a perícia. |
| PacientePessoaId | long | Atributo que vai receber o id da pessoa que identifica o paciente. |
| MedicoPessoaId | long | Atributo que especifica através do id da pessoa o médico. |
| Descrição | string | Atributo que é responsável por guardar as observações que foram feitas na perícia realizada. |

Fonte: O autor.

**Classe Agendamento**

O quadro 11 mostra os atributos da classe Agendamento.

**Quadro 11** – Atributos da classe Agendamento

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nota** | |
| AgendamentoId | long | Atributo que identifica unicamente a classe Agendamento. |
| PessoaId | long | Atributo id da pessoa que identifica o paciente agendado. |
| UsuarioId | long | Atributo responsável por receber o id do usuário. |
| ClinicaId | long | Atributo que vai receber o id que identifica a clínica do agendamento. |
| Dt\_Agendamento | date | Atributo que especifica a data do agendamento. |
| Observacoes | string | Atributo que vai receber as observações do agendamento. |

Fonte: O autor

**Classe Clinica**

O quadro 12 mostra os atributos da classe Clinica

**Quadro 12** – Atributos da classe Clinica

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nota** | |
| ClinicaId | long | Atributo responsável por garantir a unicidade da classe. |
| Nome\_Clinica | string | Atributo que recebe o nome da clinica |
| IdCidade | int | Atributo responsável por receber o id da cidade. |
| Tel\_Clinica | string | Atributo que especifica a telefone da clinica |

Fonte: O autor

**Classe Setor**

O quadro 13 mostra os atributos da classe Setor

**Quadro 13** – Atributos da classe Setor

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nota** | |
| SetorId | long | Atributo responsável por identificar unicamente a classe |
| Descricao | string | Atributo que vai receber a descrição do setor |

Fonte: O autor.

**Classe Usuário**

O quadro 14 mostra os atributos da classe Usuário

**Quadro 14** – Atributos da classe Usuário

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nota** | |
| UsuarioId | long | Atributo responsável por identificar unicamente a classe |
| Login | string | Atributo que deve receber o Login de usuário do sistema. |
| Senha | String | Atributo responsável por receber a senha do usuário. |
| pessoaId | long | Atributo que recebe o id da pessoa que tem um usuário. |
| Perfil | int | Atributo que especifica o perfil de acesso do usuário. |

Fonte: O autor.

**Classe Endereço**

O quadro 15 mostra os atributos da classe Endereço

**Quadro 15** – Atributos da classe Endereço

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nota** | |
| EnderecoId | long | Atributo responsável por identificar unicamente a classe |
| IdCidade | int | Atributo que especifica o id da cidade do endereço. |
| Cep | string | Atributo que recebe o cep do endereço. |
| Rua | string | Atributo responsável por receber a rua. |
| Numero | int | Atributo que recebe o número do endereço. |
| Bairro | string | Atributo que especifica o bairro do endereço |

Fonte: O autor.

**Classe Cidades**

O quadro 16 mostra os atributos da classe Cidades

**Quadro 16** – Atributos da classe Cidades

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nota** | |
| IdCidade | int | Atributo responsável por identificar unicamente a classe |
| Cidade | string | Atributo que vai receber o nome da cidade. |
| IdEstado | int | Atributo que especifica o id do estado |

Fonte: O autor.

**Classe Estado**

O quadro 17 mostra os atributos da classe Estado

**Quadro 13** – Atributos da classe Estado

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nota** | |
| IdEstado | int | Atributo responsável por identificar unicamente a classe |
| Estado | string | Atributo que vai receber o nome do estado. |
| Sigla | string | Atributo responsável por receber a sigla que representa o estado. |

Fonte: O autor.

**Classe Especialidade**

O quadro 17 mostra os atributos da classe Especialidade.

**Quadro 17** – Atributos da classe Especialidade.

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nota** | |
| EspecialidadeId | long | Atributo responsável por identificar unicamente a classe |
| Descricao | string | Atributo que vai receber a descrição da especialidade |

Fonte: O autor.

**Classe Requisição**

O quadro 18 mostra os atributos da classe Requisição

**Quadro 18** – Atributos da classe Requisição

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nota** | |
| RequisicaoId | long | Atributo responsável por identificar unicamente a classe |
| PacienteId | long | Atributo que especifica o id do Paciente. |
| UsuarioId | Long | Atributo que especifica o Id do Usuário que consta na requisição. |
| IdCidadeOrigem | int | Atributo que receberá o id que representa a cidade Origem. |
| IdCidadeDestino | int | Atributo que receberá o id que representa a cidade Destino. |
| dtRequisicao | date | Atributo que recebe a data de criação da requisição |
| Observacoes | String | Atributo responsável por receber as observações feita na requisição. |

Fonte: O autor.

# 

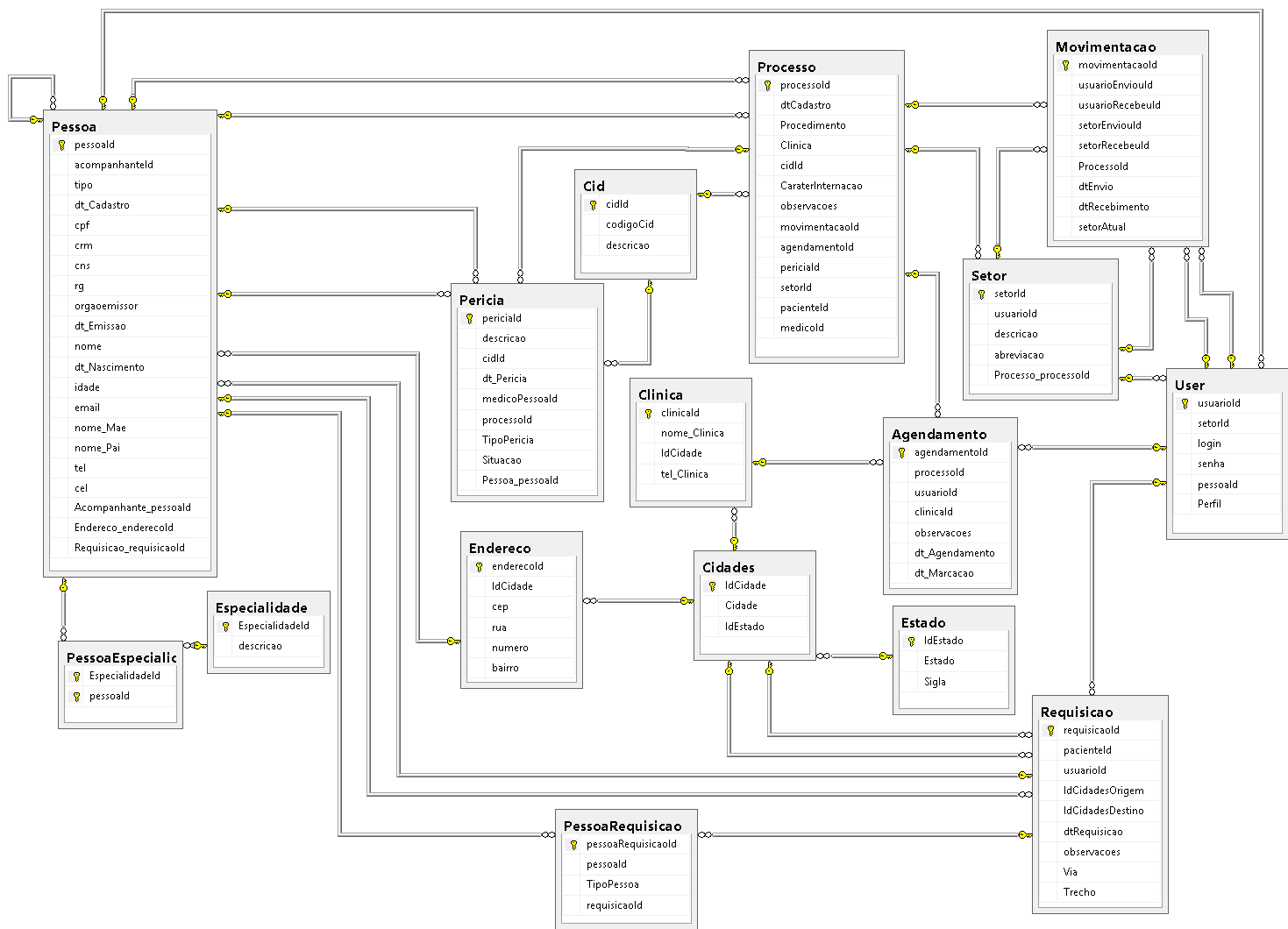
# 3.3 BANCO DE DADOS

Um banco de dados pode se traduzir como um conjunto de dados com um significado. Afirma Alves (2009, p. 23) “Um banco de dados é um conjunto lógico e ordenado de dados que possuem algum significado, e não uma coleção aleatória sem fim ou objetivo especifico. ”.

# 3.2.5 Diagrama Relacional

Para Bezerra (2007, p. 258) “O modelo Relacional baseia-se no conceito de relação, pode-se pensar em uma relação como uma tabela, composta de linhas e de colunas.”. Bezerra (2007, p. 258) Afirma ainda que “no modelo relacional, cada coluna poderá conter apenas valores atômicos, ou seja, uma coluna de uma relação não pode armazenar uma informação estruturada.”.

**Figura 19** - Diagrama Relacional



Fonte: O autor

# 4 IMPLEMENTAÇÃO

Para Implementação do SISPTD foi utilizado como ferramenta case o Enterprise Architect, para a codificação das classes do Sistema foi utilizado a linguagem de programação C# e como compilador o Microsoft Visual Studio 2013 e no auxílio do gerenciamento do banco de dados, foi utilizado Microsoft SQL Server 2012.

Na arquitetura de desenvolvimento foi empregado o MVC, MVC (Model, View, Controller), está disponível no framework ASP.NET MVC que encontra-se disponível no Microsoft Visual Studio 2013.

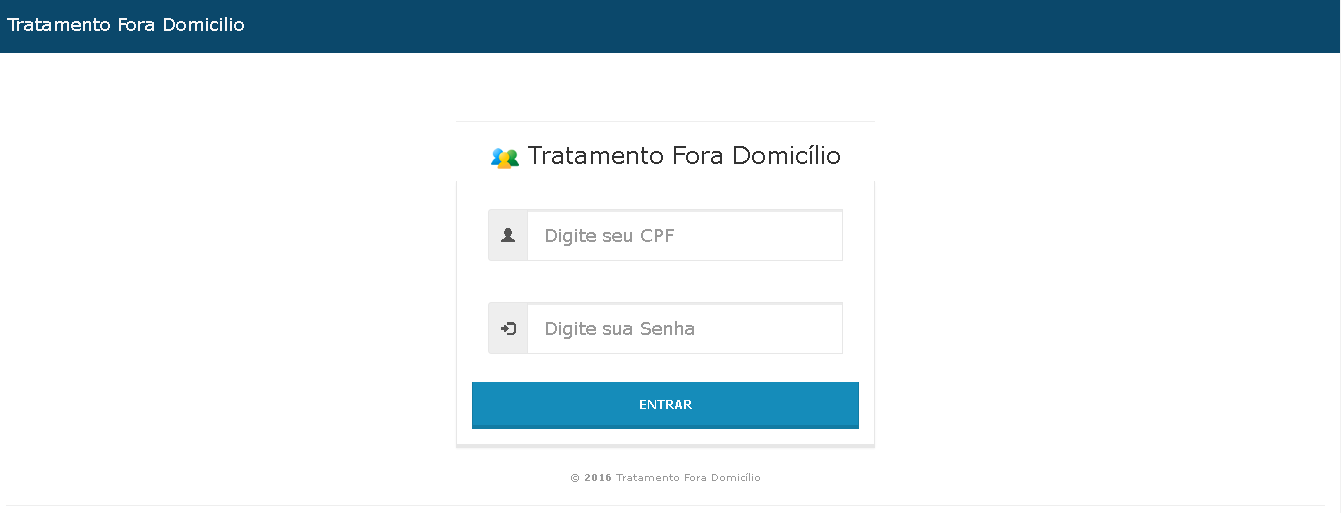
A arquitetura MVC, procura dividir as responsabilidades das partes do sistema em camadas, camadas estas que dão significado para a sigla MVC, Modelo, Visões e Controle. Onde o modelo fica responsável por deixar isolado ao restante do sistema a comunicação com as fontes de dados da aplicação, as visões tem como responsabilidade a apresentação do processamento para o usuário, ficando a cargo do controle o gerenciamento dessa comunicação, como mostraram Gamma et al. (2000, p. 20) “ O Modelo é o objeto da aplicação, a Visão é a apresentação na tela e o Controlador é o que define a maneira como a interface do usuário reage às entradas do mesmo”.

# 4.1 CAMADA DE APRESENTAÇÃO

A camada de apresentação ou interface tem por objetivo facilitar a experiência de comunicação do usuário com o sistema, é através dela que o usuário visualiza o resultado processado.

Engholm Junior (2015, p. 231) mostra que, “utilizando-se das interfaces, o usuário realiza suas entradas e manipula o sistema, sendo que nelas eles podem visualizar o estado do mesmo”. No caso da arquitetura MVC fica encargo da view a responsabilidade de renderizar as marcações HTML de formação da interface do sistema. Engholm Junior (2015, p. 231) afirma ainda que, “esse tipo de arquitetura permite que a interface da aplicação possa ser modificada sem preocupações com alterações na cada de lógica do negócio”.

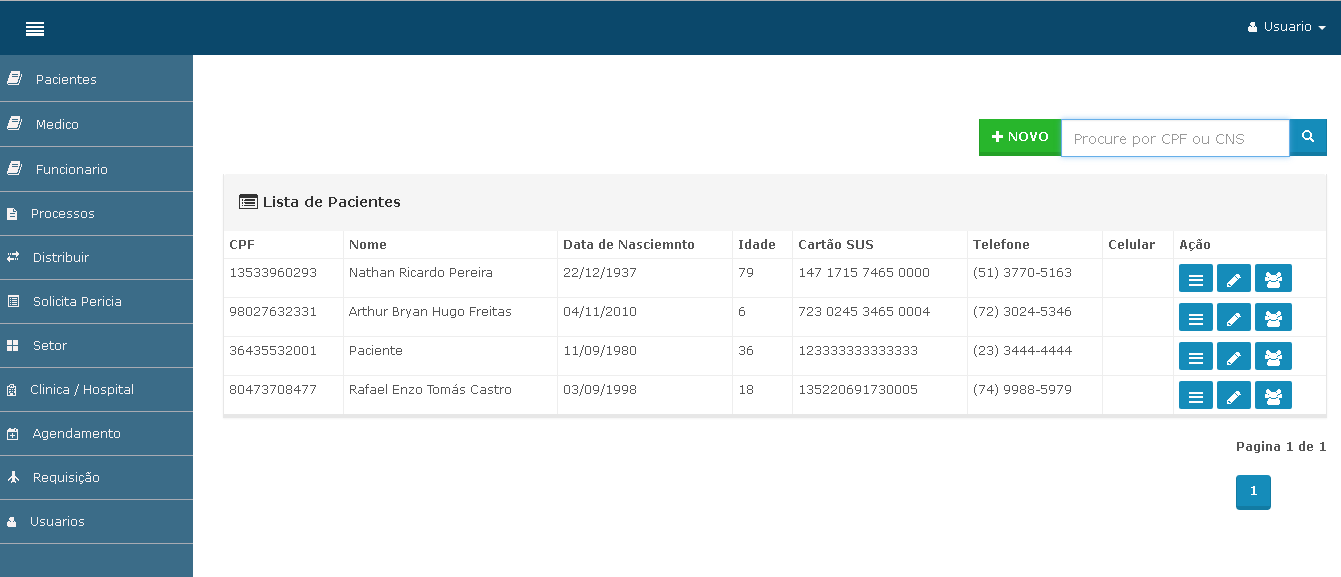
Figura 20 - Tela de Login



Fonte: O auto.

Após a realização do login, atendendo um usuário de perfil funcionário, é apresentado à tela lista de Paciente do sistema, onde é mostrado as opções do menu, conforme a figura 21.

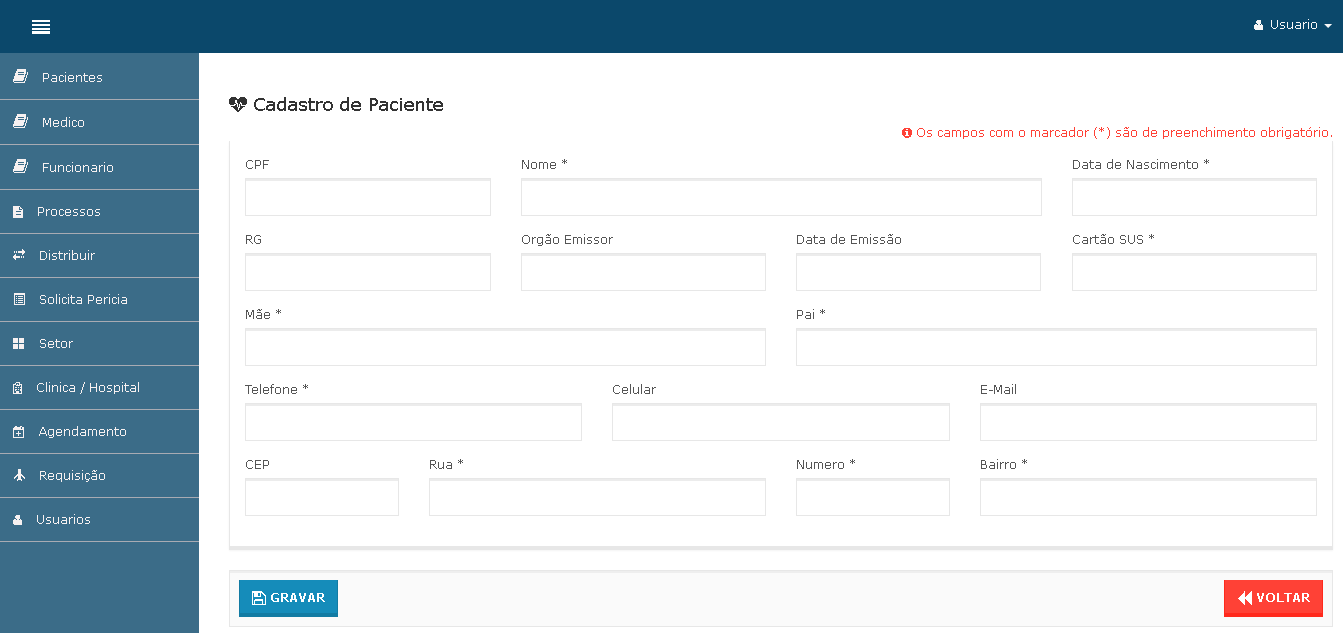
Figura 21 - Lista de Pacientes



Fonte: O autor.

Considerando o clique do usuário na opção novo, será redirecionado a tela de cadastro novo Paciente conforme figura 22.

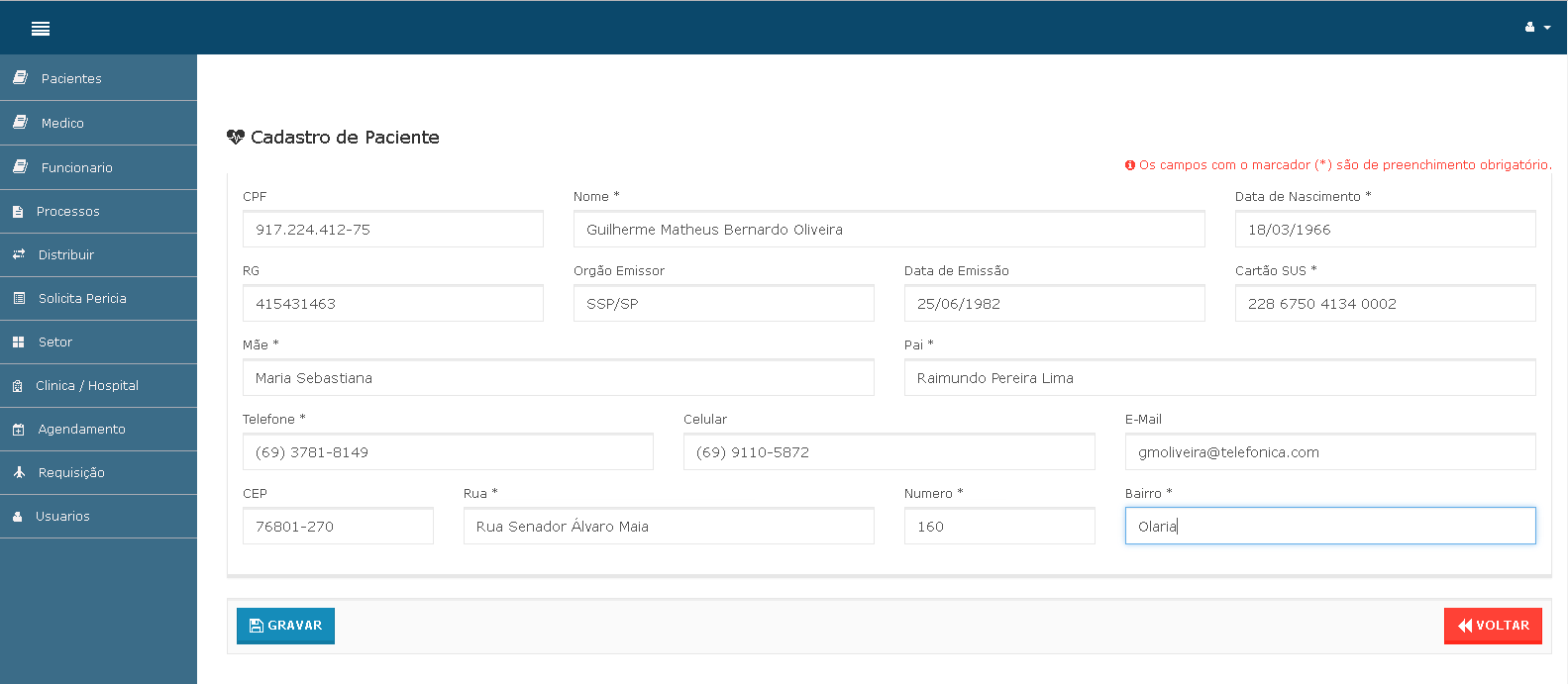
Figura 22 - Tela Cadastro de Paciente



Fonte: O autor.

Após o preenchimento dos campos com os dados referentes ao novo paciente e clicar no “Gravar” para gravar as informações, conforme figura 23.

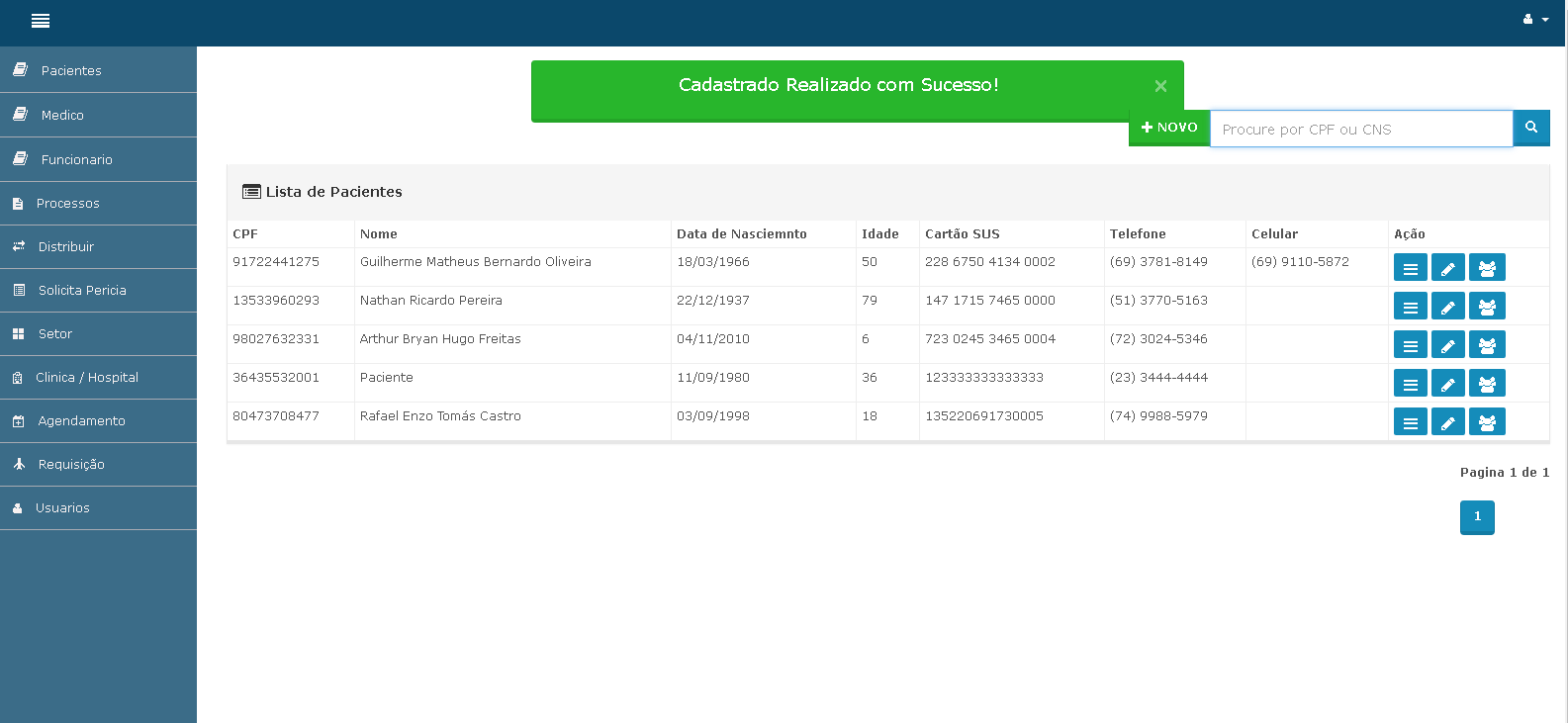
Figura 23 – Tela de cadastro de Paciente Preenchida.



Fonte: O autor.

Após o clique no botão “Gravar” o sistema redireciona o usuário para tela lista de pacientes com a mensagem “Cadastro Realizado com Sucesso!”, conforme exibe a figura 24.

Figura 24 - Tela de Pacientes com mensagem de Sucesso.



Fonte: O autor.

# 4.2 CAMADA DE CODIFICAÇÃO DE CLASSES

Adotando-se o padrão MVC, temos como camada de codificação as classes controllers e Models, ficando a encargo do controller o gerenciamento das requisições feitas pelas views. Palermo, Scheirman e Bogard (2010, p. 76) mostram que: “o foco do padrão Modelo-View-Controlador é o controlador. Com esse padrão, toda requisição é tratada por um controlador e exibida por uma view.”.

As classes controllers do sistema SISPTD interagem com uma camada denominada BO, é nessa camada que temos as classes que tratam as regras de negocio no sistema, as regras de negócio tem como finalidade tratar restrições do negócio que possam surgi durante uma requisição vinda de uma view através de um controller, essas restrições são definas no escopo das classes BO. Segundo Camacho Júnior (2008, p. 77) “as regras de negócio definem como o seu negócio funciona”. As classes da camada de negócio como é conhecida às classes BOs, se relacionam com a base de dados do sistema através de uma ferramenta de mapeamento Objeto-Relacional (ORM), Mostarda, Sanctis e Bochicchio (2012, p. 33) mostram que: “a ferramenta de O/RM fica entre o código do aplicativo e a base dados e cuida eficientemente da recuperação de dados e de sua transformação em objetos.”. Citam também que a ferramenta ORM, “rastreia as alterações nos objetos, e as reflete na base dados.”.

Como ferramenta ORM, no sistema SISPTD foi utilizado o Entity Framework (EF), que como qualquer ferramenta de mapeamento objeto-relacional procura traduzir os objetos do sistema para o banco de dados. Lerman (2010, p. 1, tradução nossa) afirma que: “o Entity Framework faz o trabalho necessário de traduzir seus objetos de volta para as linhas e colunas da tabela relacional.”.

Segue abaixo a classe pessoa que representa uma das classes pertencentes ao modelo:

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

/// <summary>

/// Classe responsável pelos dados de Pessoa

/// </summary>

public class Pessoa

{

public Pessoa()

{

}

[Key]

public long pessoaId { get; set; }

public long? acompanhanteId { get; set; }

public int? tipo { get; set; }

[Display(Name = "Data Cadastro")]

public DateTime? dt\_Cadastro { get; set; }

[StringLength(25)]

[Display(Name = "CPF")]

public string cpf { get; set; }

[StringLength(8)]

[Display(Name = "Crm")]

public string crm { get; set; }

[Required]

[StringLength(25)]

[Display(Name = "Cartão SUS")]

public string cns { get; set; }

[StringLength(25)]

[Display(Name = "RG")]

public string rg { get; set; }

[StringLength(10)]

[Display(Name = "Orgão Emissor")]

public string orgaoemissor { get; set; }

[Display(Name = "Data de Emissão")]

[DisplayFormat(DataFormatString = "{0:dd/MM/yyyy}", ApplyFormatInEditMode = true)]

public System.DateTime? dt\_Emissao { get; set; }

[Required(ErrorMessage = "Campo de Preenchimento Obrigatório")]

[StringLength(160)]

[Display(Name = "Nome")]

public string nome { get; set; }

[Display(Name = "Data de Nascimento")]

[DisplayFormat(DataFormatString = "{0:dd/MM/yyyy}", ApplyFormatInEditMode = true)]

public System.DateTime dt\_Nascimento { get; set; }

[Display(Name ="Idade")]

public int? idade { get; set; }

[StringLength(50)]

[Display(Name = "E-Mail")]

[DataType(DataType.EmailAddress)]

public string email { get; set; }

[StringLength(150)]

[Display(Name = "Mãe")]

public string nome\_Mae { get; set; }

[StringLength(150)]

[Display(Name = "Pai")]

public string nome\_Pai { get; set; }

[StringLength(20)]

[Display(Name = "Telefone")]

[Required(ErrorMessage = "Campo de Preenchimento Obrigatório")]

[DataType(DataType.PhoneNumber)]

public string tel { get; set; }

[StringLength(20)]

[Display(Name = "Celular")]

[DataType(DataType.PhoneNumber)]

public string cel { get; set; }

public virtual ICollection<Processo> ListaDeProcessosPaciente { get; set; }

public virtual ICollection<Processo> ListaDeProcessosMedico { get; set; }

public virtual ICollection<Pericia> PericiaPaciente { get; set; }

public virtual ICollection<Pericia> PericiaMedico { get; set; }

public virtual Pessoa Acompanhante { get; set; }

public virtual ICollection<User> Usuarios { get; set; }

public virtual ICollection<Especialidade> Especialidade { get; set; }

public virtual ICollection<Requisicao> RequisicaoComoPaciente { get; set; }

public virtual List<PessoaRequisicao>PessoaRequisicao { get; set; }

public virtual Endereco Endereco { get; set; }

}

# Abaixo é mostrado o código da camada BO que contem as regras de negócio para manter uma Pessoa:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using SISPTD.Models;

using System.Data.Entity;

using PagedList.Mvc;

using PagedList;

namespace SISPTD.BO

{

public class PessoaBO : CrudComumEntity<Pessoa, long>

{

public PessoaBO(dbSISPTD contexto)

: base(contexto)

{

}

public override void Inserir(Pessoa entidade)

{

if (!Ultis.Util.ValidarCPF(entidade.cpf))

throw new Exception("CPF Informado não é valido!");

if (ExistPessoa(entidade))

throw new Exception("já existe esse CPF cadastrado para uma Pessoa !");

base.Inserir(entidade);

}

public IEnumerable<Pessoa> ObterPessoa(string busca, int? pagina, int tamanhoPagina)

{

try

{

IEnumerable<Pessoa> listapessoa = \_contexto.Set<Pessoa>()

.Include(d => d.ListaDeProcessosPaciente)

.Where(x => x.cpf.Contains(busca) || x.cns.Contains(busca)).Where(x => x.tipo == 0);

return listapessoa.OrderByDescending(p => p.dt\_Cadastro).ToPagedList(pagina.Value, tamanhoPagina);

}

catch (Exception e)

{

throw new Exception("Erro na busca na lista de pessoa", e);

}

}

public IEnumerable<Pessoa> ObterMedico(string busca, int? pagina, int tamanhoPagina)

{

try

{

IEnumerable<Pessoa> listapessoa = \_contexto.Set<Pessoa>()

.Include(d => d.ListaDeProcessosPaciente)

.Where(x => x.cpf.Contains(busca)).Where(x => x.tipo == 2);

return listapessoa.OrderByDescending(p => p.dt\_Cadastro).ToPagedList(pagina.Value, tamanhoPagina);

}

catch (Exception e)

{

throw new Exception("Erro na busca na lista de Medico", e);

}

}

public IEnumerable<Pessoa> ObterFuncionarios(string busca, int? pagina, int tamanhoPagina)

{

try

{

IEnumerable<Pessoa> listapessoa = \_contexto.Set<Pessoa>()

.Include(d => d.ListaDeProcessosPaciente)

.Where(x => x.cpf.Contains(busca)).Where(x => x.tipo == 3);

return listapessoa.OrderByDescending(p => p.dt\_Cadastro).ToPagedList(pagina.Value, tamanhoPagina);

}

catch (Exception e)

{

throw new Exception("Erro na busca na lista de Funcionario", e);

}

}

public bool ExistPessoa(Pessoa pessoa)

{

try

{

var Exist = \_contexto.Set<Pessoa>().FirstOrDefault(p => p.cpf == pessoa.cpf);

if (Exist != null)

return true;

else

return false;

}

catch (Exception)

{

throw new Exception(" Outra coisa Qualquer !");

}

}

public bool CalculoIdade(Pessoa pessoa)

{

try

{

int anoNascimento = pessoa.dt\_Nascimento.Year;

int anoAtual = DateTime.Now.Year;

int idade = anoAtual - anoNascimento;

pessoa.idade = idade;

if (idade < 18 || idade > 60)

return true;

else

{

return false;

}

}

catch (Exception e)

{

throw new Exception("Erro durante a Verificação da Idade de Pessoa", e);

}

}

public string ObterPessoaLogin(User login)

{

try

{

var loginNome = Selecionar().FirstOrDefault(p => p.cpf == login.login).ToString();

return loginNome;

}

catch (Exception)

{

throw;

}

}

}

}

Abaixo Segue o código da classe PessoaController que aborda as requisições referente a pessoa:

using SISPTD.BO;

using SISPTD.Models;

using SISPTD.Ultis;

using System;

using System.Data;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Web.Mvc;

namespace SISPTD.Controllers

{

public class PessoaController : Controller

{

private PessoaBO pessoaBO = new PessoaBO(new dbSISPTD());

private CidadeBO cidadeBO = new CidadeBO(new dbSISPTD());

public ActionResult Index(int? pagina, string busca = "")

{

int tamanhoPagina = 10;

int numeroPagina = pagina ?? 1;

busca = Util.RemoverMascara(busca);

return View(pessoaBO.ObterPessoa(busca, numeroPagina, tamanhoPagina));

}

/// <summary>

/// Busca um Paciente quando passado um CPF

/// </summary>

/// <param name="cpf">string cpf</param>

/// <returns>Json</returns>

public ActionResult Pesquisar(string cpf)

{

cpf = Util.RemoverMascara(cpf);

var pessoa = pessoaBO.Selecionar().Where(p => p.cpf == cpf).FirstOrDefault();

if (pessoa == null)

{

return Json(new { Nome = "", Id = 0 }, JsonRequestBehavior.AllowGet);

}

else

{

return Json(new { Nome = pessoa.nome, Id = pessoa.pessoaId, Cpf = pessoa.cpf }, JsonRequestBehavior.AllowGet);

}

}

public ActionResult PesquisarMedico(string cpf)

{

cpf = Util.RemoverMascara(cpf);

var pessoa = pessoaBO.Selecionar().Where(p => p.cpf == cpf && p.tipo == 2).FirstOrDefault();

if (pessoa == null)

{

return Json(new { Nome = "", Id = 0, Cpf = "", Cns = "", Tel = "", Cel = "", Crm = "" }, JsonRequestBehavior.AllowGet);

}

else

{

return Json(new { Nome = pessoa.nome, Id = pessoa.pessoaId, Cpf = pessoa.cpf, Tel = pessoa.tel, Cel = pessoa.cel, Crm = pessoa.crm, Cns = pessoa.cns }, JsonRequestBehavior.AllowGet);

}

}

# 4.3 TESTE

Para que se obtenha uma ótima qualidade de software, é necessário a realização de testes para que seja colocado a prova todos os requisitos do sistema.

Engholm Júnior (2015, p. 285) afirma que: “O processo de teste de software tem por objetivo viabilizar a execução de atividades relacionadas a teste de componentes de software de aplicação.”. Já para: Sommerville (2007, p. 355) “o objetivo do estágio de teste de componente de descobrir defeitos por meio de testes de componentes individuais do programa.”.

O teste aqui utilizado foi o teste de releases ou caixa-preta onde se procura testar as especificações do sistema, de acordo com: Sommerville (2007, p. 359) “o sistema é tratado como uma caixa-preta, cujo comportamento pode ser somente determinado por meio do estudo de suas entradas e as saídas relacionadas.”, Sommerville mostra ainda que: “outro nome para isso é teste funcional, pois o testador concentra-se na funcionalidade, e não na implementação do software.”.

**Teste Funcional – Manter Paciente**

No quadro 8 é exibido as informações das atividades seguidas para a execução do teste no caso de uso Manter Paciente

**Quadro 8** – Teste Funcional – Manter Paciente

|  |  |
| --- | --- |
| **[CT001] Manter Paciente** | |
| **Descrição:** Este caso de teste tem por objetivo testar funcionalidade de interação para o cadastro de um paciente no sistema. | |
| **Requisito(s)** **Associado(s):** Efetuar Login, | |
| **Estratégia de Teste:** Funcionalidade. | |
| **Pré – Condições:**  -Estar autenticado no sistema;  -Ter perfil de funcionário | **Pós - Condições:** |
| **Critérios de Sucesso**  - Todas as funcionalidades testas na tela funcionando corretamente  -Cada uma das funções disponíveis na tela funcionando Corretamente. | |
| **Procedimento(s) Associado(s):** | |
| **Planilha(s) de Teste Associada(s):** Manter Paciente | |

Fonte: O autor

**Passos Específicos**

1. Efetuar Login no sistema;
2. Selecionar a opção “Paciente” no menu;
3. Selecionar a opção “ Novo”;
4. Preencher as informações solicitadas na tela;
5. Clicar no botão “Gravar” e verificar se foi redirecionado para tela de lista de paciente que exibirá a mensagem “Cadastro realizado com sucesso!”.

**Planilha de Teste – Manter Paciente**

Cenário de Sucesso

O quadro 9 mostra o cenário de sucesso do teste que foi realizado no caso de uso Manter Cliente.

**Quadro 9** – Planilha de Teste- Sucesso Manter Paciente

|  |  |  |  |
| --- | --- | --- | --- |
| **Entradas** | | **Saídas** | |
| **Cenário** | **Descrição** | **Resultado esperado** | **Status do Resultado** |
| CS001 | Logar no sistema | Realizar o login e aparecer a tela lista de paciente com o menu do sistema | ok |
| CS002 | Clicar na opção “Novo” | Apresentação de tela de inserção dos dados do paciente | ok |
| CS003 | Preenchimento dos campos | Aceitação do preenchimento dos campos | ok |
| CS004 | Clicar no botão “Gravar” | Redirecionamento para tela lista de paciente com a mensagem “Cadastro Realizado com Sucesso” | ok |

Fonte: O autor.

Cenário de Insucesso

O quadro 10 exibe o cenário de insucesso do teste realizado no caso de uso Manter Paciente.

**Quadro 10** – Planilha de Teste- Insucesso Manter Paciente

|  |  |  |  |
| --- | --- | --- | --- |
| **Entradas** | | **Saídas** | |
| **Cenário** | **Descrição** | **Resultado esperado** | **Status do Resultado** |
| CIS001 | Gravar um paciente sem ter preenchido todos os campos obrigatórios | Exibe uma mensagem de erro logo abaixo de cada campo não preenchido dizendo que o campo é obrigatório | ok |
| CIS002 | Clicar na opção “Novo” | Apresentação de tela de inserção dos dados do paciente | ok |
| CIS003 | Preenchimento dos campos | Aceitação do preenchimento dos campos | ok |
| CIS004 | Clicar no botão “Gravar” | Redirecionamento para tela lista de paciente com a mensagem “Cadastro Realizado com Sucesso” | ok |

Fonte: O autor.

**Teste Funcional – Requisição de Passagem**

No quadro 11 mostra as informações dos procedimentos seguidos para a execução do teste no caso de uso Requisição de Passagem

**Quadro 11** – Teste Funcional – Requisição de Passagem

|  |  |
| --- | --- |
| **[CT002] Requisição de Passagem** | |
| **Descrição:** Este caso de teste tem por objetivo testar funcionalidade de interação na realização de uma requisição de passagem. | |
| **Requisito(s)** **Associado(s):** Efetuar Login, | |
| **Estratégia de Teste:** Funcionalidade. | |
| **Pré – Condições:**  -Estar devidamente autenticado no sistema;  -Ter um dos perfis listados;  - Funcionário,  - Gerente,  - Administrador. | **Pós - Condições:**  -Emissão de uma requisição de passagem |
| **Critérios de Sucesso**  - Todas as funcionalidades testas na tela estão funcionando corretamente  -Cada uma das funções disponíveis na tela estão funcionando Corretamente. | |
| **Procedimento(s) Associado(s):** | |
| **Planilha(s) de Teste Associada(s):** Requisição de Passagem | |

Fonte: O autor.

**Passos Específicos**

1. Efetuar Login no sistema;
2. Selecionar a opção “Requisição” no menu;
3. Clicar no botão “Criar”;
4. Preencher as informações solicitadas na tela;
5. Clicar no botão “Gravar” e verificar se foi redirecionado para tela de lista de requisições que exibirá a mensagem “Cadastro realizado com sucesso!”.

**Planilha de Teste – Requisição de Passagem**

Cenário de Sucesso

O quadro 12 mostra o cenário de sucesso do teste que foi realizado no caso de uso Requisição de Passagem.

**Quadro 12** – Planilha de Teste - Sucesso Requisição de Passagem

|  |  |  |  |
| --- | --- | --- | --- |
| **Entradas** | | **Saídas** | |
| **Cenário** | **Descrição** | **Resultado esperado** | **Status do Resultado** |
| CS001 | Logar no sistema | Realizar o login e aparecer a tela lista de paciente com o menu do sistema | ok |
| CS002 | Clicar na opção “Requisição” | Apresentação de tela de lista de requisições | ok |
| CS003 | Clicar no Botão “Criar” | Apresentação do formulário a ser preenchido para gerar a requisição | ok |
| CS004 | Clicar no botão “Gravar” | Redirecionamento para tela lista de requisições com a mensagem “Cadastro Realizado com Sucesso” | ok |

Fonte: O autor.

Cenário de Insucesso

O quadro 13 exibe o cenário de insucesso do teste realizado no caso de uso Requisição de Passagem.

**Quadro 13** – Planilha de Teste- Insucesso Requisição de Passagem

|  |  |  |  |
| --- | --- | --- | --- |
| **Entradas** | | **Saídas** | |
| **Cenário** | **Descrição** | **Resultado esperado** | **Status do Resultado** |
| CIS001 | Gravar uma requisição sem ter preenchido todos os campos obrigatórios | Exibe uma mensagem de erro dizendo que o campo é obrigatório | ok |
| CIS002 | Tentar inserir um acompanhante que não seja do paciente na requisição | Exibe uma mensagem de erro dizendo “Acompanhante não pertence a esse paciente” | ok |
| CIS003 | Tentativa de inserção de mais de três acompanhantes na requisição | Exibe uma mensagem de erro alertando que só se permite até três acompanhantes | ok |

Fonte: O autor.

# 5 CONCLUSÃO

Neste capítulo você fará um breve resumo do trabalho.

Você deve apresentar qual foi a grande conclusão do trabalho.

Referir se concretizaram ou não todos os objetivos ou se não foi possível concretizar algum deles e explicar porquê.

Referir a importância que o trabalho tem para você, para o conhecimento ou aprofundamento do tema, para o seu crescimento pessoal.

# REFERÊNCIAS

Para a elaboração das referências observe a Norma ABNT NBR 6023.

Exemplos:

**Livro:**sobrenome do autor em caixa alta, nome do autor, título em negrito, edição, cidade, editora e ano de publicação. Exemplo:

PELCZAR JUNIOR, J. M. **Microbiologia:**conceitos e aplicações. 2. ed. São Paulo: Makron Books, 1996.

**Site:** sobrenome do autor, nome do autor, título do texto, ano, link e data de acesso.

Exemplo:

MORETTI, Isabella. **Regras da ABNT para TCC**: conheça as principais normas. 2014. Disponível em: <http://viacarreira.com/regras-da-abnt-para-tcc-conheca-principais-normas>. Acesso em: 02 fev. 2015.

# APÊNDICE