UNIVERSIDADE PAULISTA – UNIP EaD

Projeto Integrado Multidisciplinar – IV

Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas

VINÍCIUS MIRANDA CAMPELO – 0598532
VALERIA DA ROCHA CUNHA – 0596305

SISTEMA EM C PARA CADASTRAR PACIENTES DIAGNOSTICADOS COM A COVID-19

BRASÍLIA, 2020

VINÍCIUS MIRANDA CAMPELO – 0598532
VALERIA DA ROCHA CUNHA – 0596305

SISTEMA EM C PARA CADASTRAR PACIENTES DIAGNOSTICADOS COM A COVID-19

Projeto Integrado Multidisciplinar – IV

Projeto Integrado Multidisciplinar para obtenção do título de tecnólogo em Analise e desenvolvimento de Sistemas, apresentado à Universidade Paulista – UNIP EaD.

Orientador(a): Marcelo Santos.

BRASÍLIA, 2020

RESUMO

O Sistema em linguagem C para cadastrar pacientes diagnosticados com a COVID-19 consiste em auxiliar a equipe médica e ambulatorial a diagnosticar e cadastrar facilmente pacientes que apresentam os sintomas da Covid19. Esse sistema faz com que as informações de pacientes diagnosticados com comorbidade e pertencentes ao grupo de risco "maiores de 65 anos" sejam cadastrados em um arquivo separado com CPF e CEP para uma futura análise da secretaria de saúde da cidade. O Sistema guarda em outro arquivo os dados complementares para serem consultados pela equipe e mesmo quando o paciente apresenta sintomas mas sem comorbidades e pertencentes ao grupo de risco (idade acima de sessenta e cinco anos). O sistema foi desenvolvido em linguagem C estruturado de fácil visualização e no formato prompt de comando "DOS". Seu executável que é facilmente aberto em sistemas operacionais Windows e Linux favorece outros setores da saúde. Sua estrutura e bem simples seguindo o fluxo abordado no material de apoio. O sistema demostra confiabilidade e foi ajustado de acordo com a análise proposta.

Palavras-chave: COVID19. linguagem C. Paciente. Grupo de Risco. Sistema.

ABSTRACT

The C-language system for registering patients diagnosed with COVID-19 consists of assisting the medical and outpatient staff to easily diagnose and register patients with Covid symptoms19. This system makes the information of patients diagnosed with comorbidity and belonging to the risk group “older than 65 years” to be registered in a separate file with CPF and CEP for a future analysis by the city health department. The system stores the complementary data in another file to be consulted by the team and even when the patient has symptoms but without comorbidities and belonging to the risk group (age over sixty-five years). The system was developed in structured C language for easy viewing and in the "DOS" command prompt format. Its executable that is easily opened on Windows and Linux operating systems favors other sectors of health. Its structure is very simple, following the flow covered in the support material. The system shows reliability and has been adjusted according to the proposed analysis.

Keywords: COVID19. C. Patient language. Group of risk. System.

LISTA DE FIGURAS

Figura I. Diagrama de Caso de Uso, Lucidchart. Arquivo Próprio.....9

Figura II. Fluxograma, Lucidchart. Arquivo Próprio.....10

Figura III. Função CadastraPaciente, Notepad++. Arquivo Próprio.....11

Figura IV. Struct DadosPaciente, Notepad++. Arquivo Próprio.....12

Figura V. Função GravaCepIdade, Notepad++. Arquivo Próprio.....13

Figura VI. Função CalculaIdade, Notepad++. Arquivo Próprio.....13

Figura VII. Função FazerLogin, Notepad++. Arquivo Próprio.....14

Figura VIII. Função ConsultaPaciente, Notepad++. Arquivo Próprio.....15

SUMÁRIO

1. INTRODUÇÃO.....6
2. OBJETIVO.....7
2.1. OBJETIVO GERAL.....7
2.2. OBJETIVO ESPECÍFICOS.....7
2.3. MOTIVAÇÃO E CONTEXTUALIZAÇÃO.....8
2.4. DESENVOLVIMENTO.....11
3. CONCLUSÃO.....16
4. REFERÊNCIAS.....17

1. INTRODUÇÃO

Segundo (MS, 2020) a COVID-19 é uma doença causada pelo coronavírus, denominado SARS-CoV-2, que apresenta um espectro clínico variando de infecções assintomáticas a quadros graves. De acordo com a Organização Mundial de Saúde, a maioria (cerca de 80%) dos pacientes com COVID-19 podem ser assintomáticos ou oligossintomáticos (poucos sintomas), e aproximadamente 20% dos casos detectados requer atendimento hospitalar por apresentarem dificuldade respiratória, dos quais aproximadamente 5% podem necessitar de suporte ventilatório.

No Brasil os casos confirmados chegam a quase 5.600.00 no seu acumulado estão chegando a 5.070.000 de casos recuperados e isso leva a uma grande preocupação de nossos governantes. Cada estado brasileiro cria uma politica para frear esse vírus mortal e em muitos casos são criados sistemas para verificar como esse vírus se alastra pelo país. O ministério da Saúde criou um sistema de estatística para monitoramento do vírus nas grandes capitais com a ajuda de outros órgãos da saúde como hospitais e secretarias de saúde. O governo vem fazendo um estudo prévio usando geoprocessamento e criando sistemas internos para registrar onde há mais casos confirmados em qual local esses pacientes estão sendo tradados.

Hoje em dia estão utilizando novas tecnologias para criação de sistemas de avaliação e isso não altera caso use uma linguagem antiga para criação de sistemas. Uma boa linguagem de programação utilizada e a linguagem C e C++ esta última orientada a objetos e usada ate os dias atuais.

A linguagem C de acordo com (SCHILDT, 1996) é frequentemente chamada de linguagem de médio nível para computadores. Ela é tratada assim porque combina elementos de linguagem de alto nível com a funcionalidade da linguagem assembly. Uma linguagem estruturada permite muitas possibilidades na programação; Ela suporta, diretamente, diversas construções de laços (loops), como while, do-while e for. A linguagem C foi inventada e implementada por Dennis Ritchie na década de 1970 e atualmente e usada na comunidade acadêmica por ter boa didática.

2. OBJETIVO

2.1. OBJETIVO GERAL

Apresentar um sistema em C que será utilizado pelos hospitais para cadastrar os pacientes que forem diagnosticados com covid-19 e carecem de um acompanhamento e monitoramento para que essa informação possa ser enviada para a central da Secretaria da Saúde.

2.2. OBJETIVO ESPECÍFICOS

- Desenvolver e aplicar os conhecimentos adquiridos nas disciplinas de linguagem e técnicas de programação e engenharia de software I;
- Fomentar o hábito de executar projetos envolvendo múltiplas disciplinas;
- Desenvolver a capacidade de identificar necessidades e propor soluções técnicas;
- Elencar, argumentar e justificar sobre metodologias referentes ao desenvolvimento de um sistema para auxiliar a área da saúde;
- Aplicar as normas ABNT para a produção de trabalhos acadêmicos.

2.3. MOTIVAÇÃO E CONTEXTUALIZAÇÃO

A engenharia de software vem evoluindo trazendo novas técnicas e melhorando a vida do desenvolvedor e gestores e não podendo esquecer dos analistas de requisitos o qual produz toda a documentação. Nos dias atuais de quem trabalha no “front” do desenvolvimento precisa lidar com toda essa tecnologia alias, o mundo evoluiu levando consigo sistemas sofisticados e com muitos bytes. O Sistema para cadastrar pacientes diagnosticados com a COVID-19 tem a função de seguir esses paradigmas e trazer facilidade na produção. Da mesma forma ele segue as especificações de elaboração utilizando a IDE codeblocks e o desenvolvimento em linguagem “C”. Uma linguagem totalmente voltada ao publico acadêmico que tem as principais vantagens:

- Ser estruturada facilitando no aprendizado dentro e fora das universidades;
- Ser multiplataforma suportada por várias plataformas como: (Windows, Mac e Linux);
- E simples ela influenciou outras linguagens como: C++ e (C Sharp).

Infinitas possibilidades porém a linguagem não trabalha sozinha é preciso pessoas qualificadas para fazer bons projetos engenheiros de software, analistas de sistemas, programadores, designer, documentadores, analista de requisitos e o principal os stakeholders, as partes interessadas e quem decide sobre o projeto por exemplo: pessoas, organizações, direta e indiretamente engajadas no sucesso do produto.

No desenvolvimento de sistema para cadastrar pacientes diagnosticados com a COVID-19 utilizou-se de alguns processos da engenharia de software. No levantamento de requisitos foi utilizado para construção do sistema o diagrama de caso de uso e um fluxograma que auxiliou no processo de desenvolvimento.

Antes de iniciar o assunto precisamos entender o que é a Linguagem Unificada de Modelagem “UML”. Para DA SILVA e VIDEIRA (2001, p.111) “O UML é uma linguagem diagramática, utilizável para especificação, visualização e documentação de sistemas de software”. Seu surgimento foi na década de 90 no intuito de unificar as principais linguagens de modelagem orientadas por objetos. Ao utilizar o diagrama da UML no projeto foi levado em consideração a facilidade e o entendimento do processo.

Seguindo o manual do projeto integrado multidisciplinar “PIM” em sua contextualização do caso, o objetivo do projeto e desenvolver um sistema em linguagem “C” que será utilizado pelos hospitais para cadastrar os pacientes que forem diagnosticados com a covid-19 e carecem de um acompanhamento e monitoramento. Ao receber o diagnostico positivo os profissionais da saúde devem realizar o login no sistema em seguida cadastrar esses dados como nome, CPF, telefone, endereço (rua, número, bairro, cidade, estado e CEP), data de nascimento e e-mail, data do diagnostico e informa alguma comorbidade do paciente (diabetes, obesidade, hipertensão, tuberculose, outros que serão salvos em um arquivo que poderão ser consultadas a qualquer momento).

Após esse cadastro, o sistema deverá calcular a idade e verificar se o paciente possui alguma comorbidade e se pertence ao grupo de risco no caso (maiores de 65 anos). Caso pertença ao grupo de risco o sistema devera salvar em um arquivo de texto o CEP e a idade do paciente para ser enviado a central da Secretaria de Saúde da cidade. (UNIP, 2020. p.24).

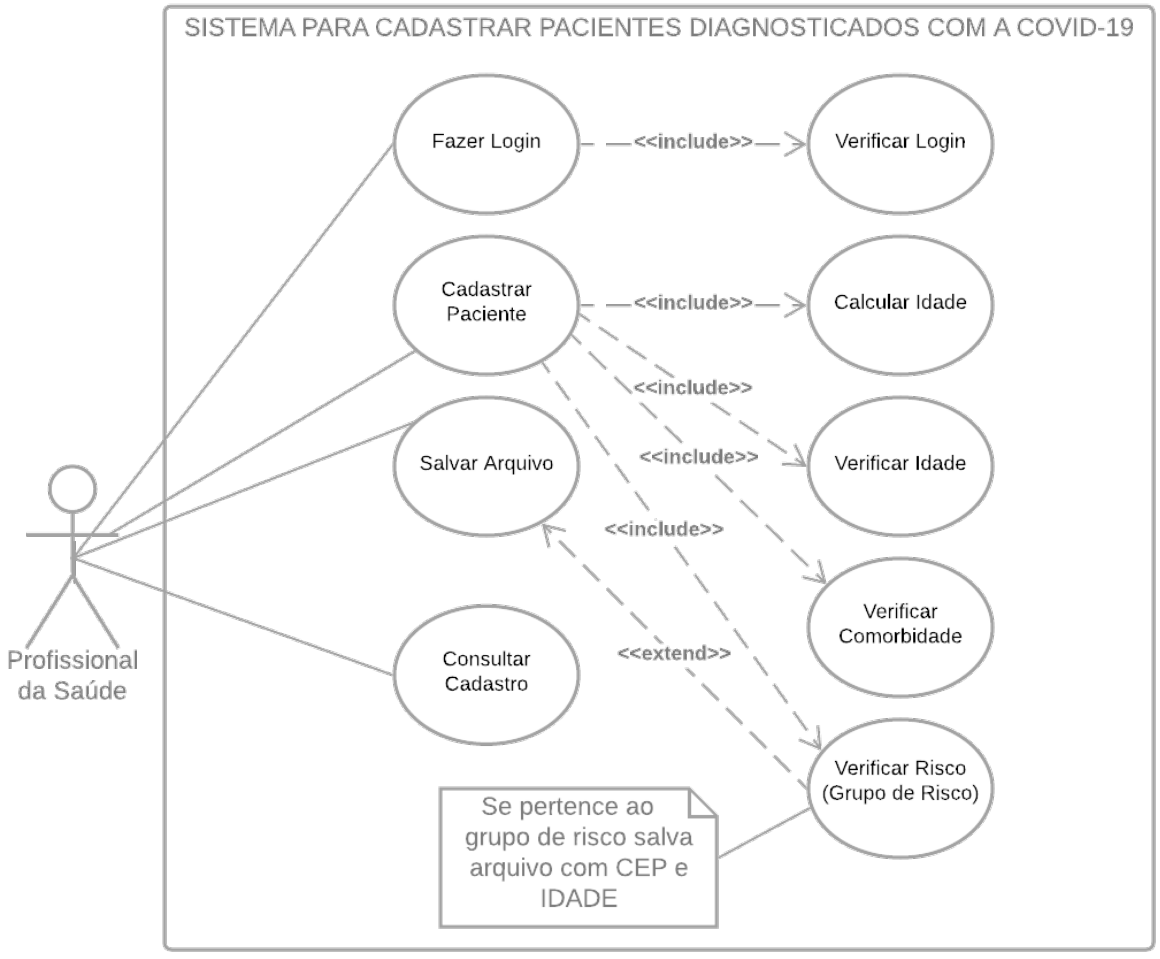

Figura I. Diagrama de Caso de Uso, Lucidchart. Arquivo Próprio.

De acordo com (VIEIRA, 2020) “O diagrama de Casos de Uso auxilia no levantamento dos requisitos funcionais do sistema, descrevendo um conjunto de funcionalidades do sistema e suas interações com elementos externos e entre si”.

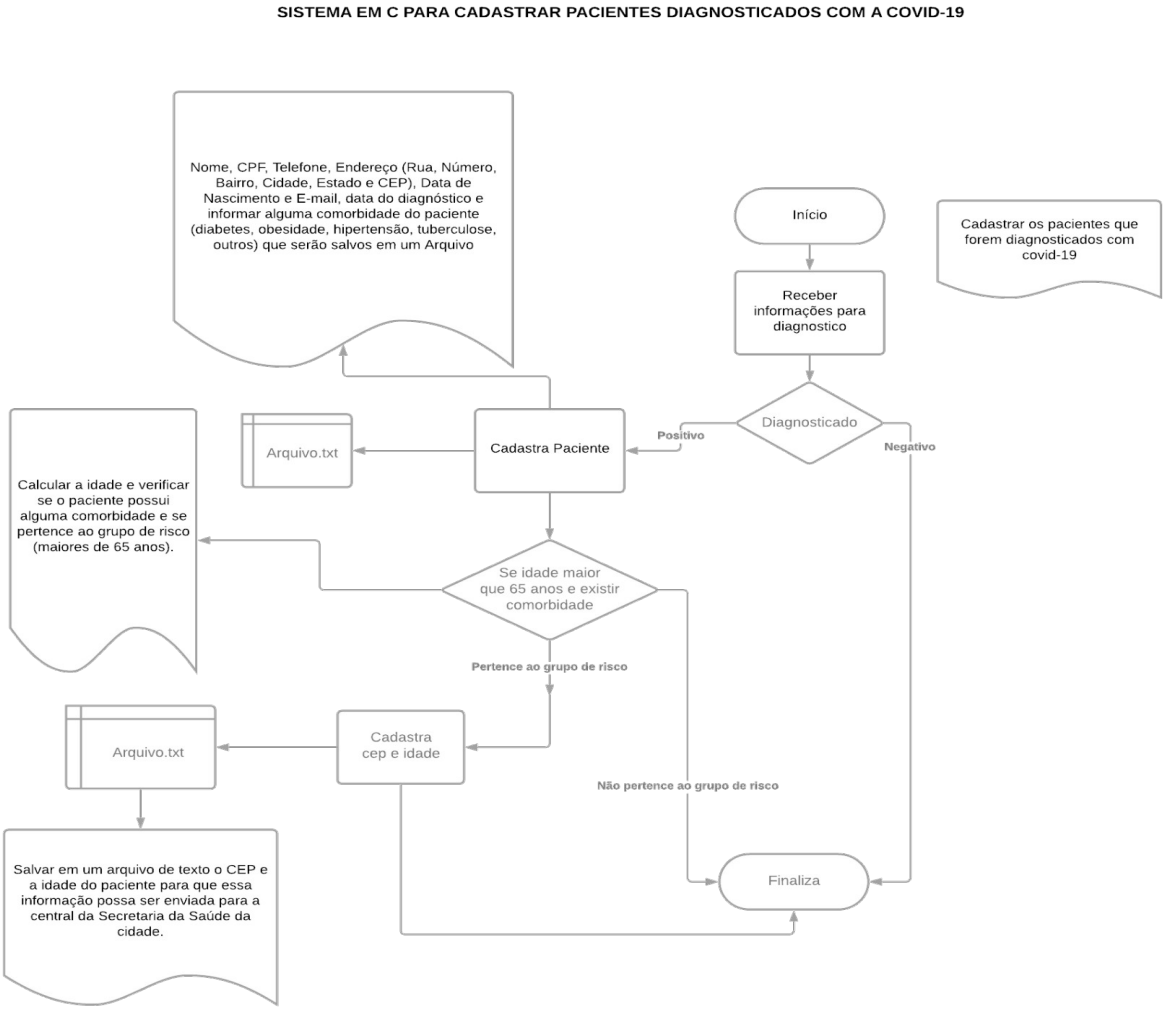

Figura II. Fluxograma, Lucidchart. Arquivo Próprio.

O principal objetivo de um fluxograma e melhorar a compreensão dos processos e a forma como eles se interligam, evidenciar como as rotinas e as atividades são desempenhadas dentro da empresa, encontrar os problemas que acarretam desperdício de tempo e de recursos (FLUXOGRAMA, 2020).

Ouvi-se falar muito de estórias mas afinal o que é uma estória na engenharia de software? De acordo com (BERNARDO, 2020) “uma estória de usuário pode ser caracterizada como uma curta e simples descrição da necessidade do cliente. Ela normalmente

é contada a partir da perspectiva de quem precisa da nova necessidade, sendo geralmente um usuário, cliente do sistema ou representante de negócios do cliente".

A estória de usuário deve ser muito bem explicada para que não haja dúvidas. Isso é importante, principalmente, para desenvolvedores que poderão construir softwares com menor dificuldade e maior qualidade, entendendo os objetivos e necessidades do cliente de forma mais rápida.

Na modelagem e elaboração desses arquivos a maior complexidade é entender o que realmente o cliente deseja porém, vamos entrar na parte do desenvolvimento do sistema e melhor compreender sua criação.

2.4. DESENVOLVIMENTO

De acordo com o fluxograma e o caso de uso será atribuído algumas partes da estrutura do projeto desenvolvidas na linguagem "C" com a metodologia adquirida na aula de engenharia de software I.

Foi criado uma classe com o nome "Paciente.h" com os principais métodos que será chamado dentro do arquivo "main.c" ela foi separada do main pois a visibilidade da estrutura ficaria comprometida.

```
/*  CADASTRAR PACIENTE
    Cadastrar todos os dados de paciente
    indiferente de ter comorbidade ou está
    no grupo de risco em arquivo.txt.
*/
bool CadastraPaciente(DadosPaciente cadastrar){

      FILE *arquivo;
      //ABRE O ARQUIVO CADASTRO_TOTAL PARA GRAVAÇÃO NO FINAL DO ARQUIVO
      //CASO TENHA ALGUMA INFORMAÇÃO
      arquivo = fopen("CADASTRO_TOTAL.txt", "ab");
        if (arquivo == NULL) {
            return false;
        }
        //CALCULA IDADE
        int idade = CalculaIdade(cadastrar);

            fprintf(arquivo, "%s,", cadastrar.nome);
            fprintf(arquivo, "%s,", cadastrar.cpf);
            fprintf(arquivo, "%s,", cadastrar.telefone);
            fprintf(arquivo, "%s NUMERO:%d. %s %s-%s,", cadastrar.endereco, cadastrar.numero, cadastrar.bairro, cadastrar.cidade, cadastrar.estado);
            fprintf(arquivo, "%s,", cadastrar.cep);
            fprintf(arquivo, "%d/%d/%d,", cadastrar.dia, cadastrar.mes, cadastrar.ano);
            fprintf(arquivo, "%s,", cadastrar.email);
            fprintf(arquivo, "%d/%d/%d,", cadastrar.dia_c, cadastrar.mes_c, cadastrar.ano_c);
            fprintf(arquivo, "%s,", cadastrar.comorbidade);
            fprintf(arquivo, "%d\n", idade);

          if(arquivo != NULL){
            fclose(arquivo);
            return true;
          }else{
            fclose(arquivo);
            return false;
    }

}
```

Figura III. Função CadastraPaciente, Notepad++. Arquivo Próprio.

No código acima foi usado o tipo FILE que está definido na biblioteca “stdio.h” e manipulado através de um ponteiro especial para o arquivo e sua função e apontar a localização de um registro. Ele recebe por parâmetro o *tipo* e *local do arquivo* e se o arquivo será leitura, escrita ou leitura e escrita ao mesmo tempo. Ao final o arquivo deve ser fechado. A função e do tipo boolean e deve retornar true ou false nessa mesma função está sendo usado uma struct “estrutura” ***DadosPaciente*** que está sendo passada por valor preenchendo e sendo salvo suas informações em um “arquivo.txt”.

A classe “Dados.h” foi criada para armazenar um pacote de variáveis no caso uma struct que após sua criação poderá ser chamada pelo seu tipo.

```
1  #ifndef DADOS_H
2  #define DADOS_H
3
4  typedef struct{
5
6      char usuario[10];
7      char senha[10];
8      char nome[40];
9      char cpf[11];
10     char telefone[15];
11     char endereco[30];
12     int numero;
13     char bairro[20];
14     char cidade[20];
15     char estado[2];
16     char cep[8];
17     int dia, mes, ano;
18     int idade;
19     char email[60];
20     int dia_c;
21     int mes_c;
22     int ano_c;
23     char comorbidade[30];
24 }DadosPaciente;
25 #endif // DADOS_H
```

Figura IV. Struct DadosPaciente, Notepad++. Arquivo Próprio.

A próxima função “GravaCepIdade” salvar em um arquivo-texto o CEP e a IDADE do paciente para que essa informação possa ser enviada para a central da Secretaria da Saúde.

```
/*
    Gravar CEP/IDADE em um arquivo separado.
*/
bool GravaCepIdade(DadosPaciente cadastrar){

    FILE *arquivo;
    arquivo = fopen("arquivo_especifico.txt", "ab");

    if (arquivo == NULL) {
        return false;
    }

    fprintf(arquivo, "CEP: %s  - IDADE: %d\n", cadastrar.cep, CalculaIdade(cadastrar));
     if(arquivo != NULL){
          fclose(arquivo);
          return true;
        }else{
          fclose(arquivo);
          return false;
    }
}
```

Figura V. Função GravaCepIdade, Notepad++. Arquivo Próprio.

A função “CalculaIdade” foi criada sem o uso de bibliotecas prontas como “ctime.h” usamos o DEFINE para armazenar o mês e ano atual que devera ser mudado manualmente ao final a função retorna a idade do paciente.

```
#define ANO_ATUAL 2020 //modifica essa linha ano atual
#define MES_ATUAL 11   //modificar essa linha mes atual
/*
  Calcula a idade atual.
*/
int CalculaIdade(DadosPaciente usuario)
{
    int _idade_aniversariante=0, _mes_aniversariante=0;

    _mes_aniversariante = MES_ATUAL + 1;
    _idade_aniversariante = ANO_ATUAL - usuario.ano;

    //verifica se o mes e maior ao mes atual;
    if(_mes_aniversariante > usuario.mes){
        usuario.idade = _idade_aniversariante;
      }else{
        usuario.idade = (_idade_aniversariante - 1);
    }
    return usuario.idade;
}
```

Figura VI. Função CalculaIdade, Notepad++. Arquivo Próprio.

Função “FazerLogin” como especificado no diagrama de caso de uso e seu usuario e senha estão diretamente gravados na função. Usa-se a função strcmp() da biblioteca string.h que compara strings quando igual a zero o conteúdo das strings são iguais assim retornando boolean para validação.

```
26  /*
27    Verifica se usuario fez login
28    com as credeciais ja cadastradas
29    Usuario:"UNIP", Senha:"1234";
30  */
31  bool FazerLogin(DadosPaciente verifica)
32  {
33      if (strcmp(verifica.usuario, "UNIP") == 0 && strcmp(verifica.senha, "1234") == 0)
34      {
35          return true;
36        }else{
37          return false;
38      }
39  }
```

Figura VII. Função FazerLogin, Notepad++. Arquivo Próprio.

```
108  //FUNCAO CONSULTAR PACIENTE
109  bool ConsultaPaciente(char consultaNome[30]){
110
111    int i=0;
112    FILE *arquivo;
113    char string[2000]={0};
114    //VETOR DE CARACTERES
115    char texto[10][30] = {"NOME","CPF","TELEFONE","ENDERECO","CEP","DATA_NASCIMENTO",
116                          "EMAIL","DATA_CADASTRO_COMORBIDADE","COMORBIDADE","IDADE"};
117
118    if(consultaNome[0] != '0')
119    {       //ABRE O ARQUIVO CADASTRO_TOTAL.TXT PARA LEITURA E RETORNO
120        if ((arquivo=fopen ("CADASTRO_TOTAL.txt", "rb")) != NULL)
121        {
122            //LER OS ARQUIVOS COM ESPACO E PULAR LINHA %[^\n]\n
123            while( (fscanf(arquivo, "%[^\n]\n", &string))!=EOF) // VERIFICA SE O FIM DO ARQUIVO
124            {
125                //strtok DEVOLVE UM PONTEIRO PARA A PROXIMA PALAVRA (DELIMITADOR)
126                char *dados_paciente = strtok(string, ",");
127                //RETORNA A POSIÇÃO DA PRIMEIRA OCORRENCIA VINDA DA PRIMEIRA CADEIA
128                //DE TEXTO DO ARQUIVO LIDO E COMPARA COM DIGITADO
129                if(strstr(string, consultaNome)!=0)
130                {
131                    while(dados_paciente != NULL)
132                    {       //IMPRIME CONFORME O DELIMITADOR E RETORNA PARA PROXIMA POSIÇÃO
133                            printf("%s: %s\n",texto[i], dados_paciente);
134                            dados_paciente = strtok(NULL, ",");
135                            i++;
136                    }
137                }
138            }
139               fclose(arquivo); //FECHA ARQUIVO
140            return true;
141        }
142         return true;
143    }
144  }
```

Figura VIII. Função ConsultaPaciente, Notepad++. Arquivo Próprio.

O código acima verifica o uso de como validar ou fazer o tratamento de algumas funções com IF e WHILE juntos. Em projetos na linguagem “C” o tratamento em muitos casos acontece na própria página “main.c”. A linguagem “C” e pouco complexa e devemos está atento as validações do sistema.

3. CONCLUSÃO

Conclui-se que, foi possível apresentar os conhecimentos adquiridos nas matérias linguagem de programação e engenharia de software I um pouco das metodologias apresentadas e entregar o sistema de acordo com as especificações do projeto integrado multidisciplinar simples e objetivo.

Com planejamento o desenvolvimento de projetos fica prático e auxilia no entendimento do processo de acordo com a engenharia de software. Deve ser observado qual linguagem usar e entender seu comportamento e funções. Tentar o máximo adquirir técnicas que facilitam o reúso e utilizar de boas praticas para um código limpo e de fácil manutenção. E importante nesse conjunto de ações documentar respeitando os padrões de cada linguagem.

4. REFERÊNCIAS

MS, Ministério da Saúde. **Coronavírus COVID19 – O que você precisa saber**. Disponível em: <https://coronavirus.saude.gov.br/>. Acesso em 09 de novembro de 2020.

SCHILDT, Herbert; **C, completo e total – 3ª edição e atualizada Hebert Schildt**. São Paulo: McGraw-Hill, Ltda e Makron Books do Brasil Editora Ltda. 1996.

DA SILVA, A, M, R; VIDEIRA, C, A, E; UML, **Metodologias e Ferramentas CASE – Linguagem de modelação UML, Metodologias e Ferramentas CASE na Concepção e Desenvolvimento de Software**. Portugal: Centro Atlântico. 2001.

UNIP, Universidade Paulista; Projeto Integrador Multidisciplinar – PIM. São Paulo: UNIP, 2020. 29 p.

BERNARDO, K; **Estória de usuário. Você saberia contar, Cultura Ágil.** Disponível em: <https://www.culturaagil.com.br/estoria-de-usuario-voce-saberia-contar/>. Acesso em: 12 de novembro. De 2020.

VIEIRA, R; **UML – Diagrama de Casos de Uso, OperacionalTI.** Disponível em: <https://medium.com/operacionalti/uml-diagrama-de-casos-de-uso-29f4358ce4d5>. Acesso em: 13 de novembro. De 2020.

Fluxograma. **Como fazer Fluxograma, Modelos de Fluxogramas Online**. Disponível em: <https://fluxograma.net/>. Acesso em 12 de novembro de 2020.