



UNIVERSIDADE PAULISTA

PROJETO INTEGRADO MULTIDISCIPLINAR

LUCAS CORREA ALVES - RA 0600580

**PIM IV - SISTEMA EM C PARA CADASTRAR PACIENTES DIAGNOSTICADOS
COM COVID-19**

Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas

UNIP - ALPHAVILLE (POLO EAD)

2022

PROJETO INTEGRADO MULTIDISCIPLINAR

LUCAS CORREA ALVES - RA 0600580

**PIM IV - SISTEMA EM C PARA CADASTRAR PACIENTES DIAGNOSTICADOS
COM COVID-19**

Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas

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

Orientador(a):

UNIP - ALPHAVILLE (POLO EAD)

2022

PROJETO INTEGRADO MULTIDISCIPLINAR
LUCAS CORREA ALVES - RA 0600580

PIM IV - SISTEMA EM C PARA CADASTRAR PACIENTES DIAGNOSTICADOS
COM COVID-19

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

UNIP - ALPHAVILLE (POLO EAD), __ de _____ de ____

BANCA EXAMINADORA

Prof. Dr.
Universidade

Prof. Dr.
Universidade

Prof. Dr.
Universidade

RESUMO

A pandemia do Covid-19 provocou uma grande mudança na vida e cotidiano de todos, o risco a infecção forçou países a fecharem fronteiras, reduzir o contato populacional e teve como principal área afetada a saúde. Muitos hospitais e postos de saúde precisaram se reinventar e desenvolver novas práticas para suportar as novas demandas da população, cada minuto se tornou ainda mais valioso. Visto isso, qualquer medida visando o incremento de desempenho nos processos do combate a pandemia se tornou rapidamente um interesse em comum.

É com este propósito que o presente projeto foi desenvolvido, mediante metodologias ágeis presentes na Engenharia de Software e utilizando conceitos de Linguagem e Técnicas de programação, nosso objetivo é facilitar e acelerar processos manuais, ampliar o controle e manuseio dos dados gerados nas consultas gerando a menor quantidade possível de erros mecânicos e outliers, permitindo uma comunicação de dados mais eficiente com a central da Secretaria da Saúde.

Palavras-chave: Covid-19

ABSTRACT

The Covid-19 pandemic provoked a great change in the daily lives of everyone, the risk of infection forced countries to close their borders, reduce population contact and had health as the main area affected. Many hospitals and health centers needed to reinvent themselves and develop new practices to support new demands of the population, every minute became even more valuable. Any measure aimed at increasing processes performance, quickly became a common interest.

With this propose, the present project was developed, through agile methodologies present in Software Engineering and using concepts of Language and Programming Techniques, our objective is to facilitate and accelerate manual processes, expand the control and handling of the data generated in the queries, generating the least amount of mechanical errors and outliers possible, allowing for more efficient data communication with the Health Department's central office.

Keywords: covid-19

SUMÁRIO

| | | |
|-------|---------------------------------------|----|
| 1 | INTRODUÇÃO | 7 |
| 2 | PROBLEMÁTICA | 8 |
| 3 | ENGENHARIA DE SOFTWARE | 9 |
| 3.1 | Metodologias de desenvolvimento | 9 |
| 3.2 | Metodologia Cascata (Waterfall) | 9 |
| 3.3 | Metodologias tradicionais | 9 |
| 3.4 | Metodologia Ágil | 10 |
| 3.5 | Quadro Kanban | 10 |
| 3.6 | Utilização prática | 10 |
| 4 | LINGUAGEM DE PROGRAMAÇÃO C | 12 |
| 4.1 | Tipos de variáveis | 12 |
| 4.2 | Estruturas de dados | 12 |
| 4.3 | Laços de repetição | 13 |
| 4.3.1 | Laço for | 13 |
| 4.3.2 | Laço While | 13 |
| 4.3.3 | Laço Do-While | 14 |
| 4.4 | Funções | 14 |
| 4.5 | Bibliotecas de funções | 15 |
| 4.6 | Listas encadeadas | 16 |
| 5 | FUNCIONAMENTO DO SOFTWARE | 18 |
| 5.1 | Estruturas desenvolvidas | 18 |
| 5.2 | Funções e Procedimentos | 19 |
| 5.3 | Menu principal | 21 |
| 5.4 | Cadastro de paciente | 22 |
| 5.5 | Alterar dados cadastrados | 22 |
| 5.6 | Consultar dados do paciente | 23 |
| 5.7 | Arquivo Texto | 24 |
| 6 | CONCLUSÃO | 26 |
| | REFERÊNCIAS | 27 |

LISTA DE ILUSTRAÇÕES

| | | |
|-------------|---------------------------------------------------------------------------------------------------|----|
| Imagem 1 — | Exemplo de uso do quadro Kanban | 10 |
| Tabela 1 — | Tipos de Dados | 12 |
| Imagem 2 — | Exemplo de estrutura de dados, tipo "pessoa". | 13 |
| Imagem 3 — | Exemplo de uso Do-while | 14 |
| Imagem 4 — | Exemplo de função na linguagem C | 15 |
| Imagem 5 — | Bibliotecas utilizadas para o desenvolvimento do Software | 16 |
| Imagem 6 — | Exemplo de funcionamento de uma lista encadeada | 17 |
| Imagem 7 — | Estrutura tipo "login". | 18 |
| Imagem 8 — | Estrutura tipo "Endereço". | 18 |
| Imagem 9 — | Estrutura tipo "log". | 19 |
| Imagem 10 — | Estrutura tipo "pessoa". | 19 |
| Imagem 11 — | Estrutura tipo "Nó". | 19 |
| Imagem 12 — | Acesso administrativo do sistema | 20 |
| Imagem 13 — | Função de consulta de usuários | 21 |
| Imagem 14 — | Execução do menu principal do software | 22 |
| Imagem 15 — | Menu de alterações | 23 |
| Imagem 16 — | Exibição dos dados consultados de um paciente fictício durante a execução do software. | 24 |
| Imagem 17 — | Arquivo "grupoderisco.txt" sendo gerado pelo software. | 25 |
| Imagem 18 — | Arquivo gerado pelo software, exemplo de relatório. | 25 |

1 INTRODUÇÃO

O século XXI ainda está em seu primeiro terço de existência, e neste breve período passamos por acontecimentos marcantes, como o avanço tecnológico, guerras e pandemias, fazendo com que a sociedade que conhecemos passe por mudanças bruscas, e a partir disso, termos como o "novo normal" recebem sua devida definição. A pandemia de Covid-19 atualmente se encontra em um período "controlado", devido aos esforços comunitários dos países do globo. Um dos motivos para atingirmos este feito, foi a aplicação de regras sanitárias e os grandes investimentos no desenvolvimento de softwares voltados para o auxílio a saúde.

O combate ao Covid-19 continua, e com isso, restam problemas a serem resolvidos, como o rastreamento dos pacientes que foram diagnosticados com o vírus, a fim de alertar os civis próximos e contribuir com a prevenção. É com este propósito que desenvolvemos um software que irá auxiliar os profissionais da saúde a controlar o fluxo de dados dos pacientes diagnosticados, automatizando o processamento de informação, evitando erros de digitação e o problema da escrita, onde em muitos casos se torna ilegível e a informação contida é perdida.

2 PROBLEMÁTICA

Nosso objetivo foi desenvolver um sistema que será utilizado pelos profissionais da saúde, em hospitais, onde os pacientes diagnosticados com o Covid-19 e que careçam de acompanhamento e monitoramento serão cadastrados. Para isso, o sistema necessita de uma função para o controle de quem acessa e possui autorização para realizar o cadastro desses pacientes, utilizando um sistema de login e senha.

Seguindo para o cadastro dos pacientes positivados, o programa deve armazenar os dados pessoais do paciente, como CPF, telefone, endereço (rua, número, bairro, cidade, estado e CEP), data de nascimento, e-mail, data do diagnóstico e, quando necessário, informar as comorbidades existentes (diabetes, obesidade, hipertensão, tuberculose etc.). Esses dados são salvos em um arquivo que fará a função de banco de dados, podendo ser consultado e atualizado a qualquer momento.

Após o cadastro, o software deve calcular a idade do paciente e verificar se o mesmo se encontra no grupo de risco, através das seguintes condições:

- Pacientes maiores de 65 anos.
- Pacientes com comorbidades.

Qualquer das condições citadas, fará o paciente ser enquadrado no grupo de risco, e seus dados de CEP e idade precisam ser enviados para a central da Secretaria da Saúde, onde esses dados serão utilizados para aprimorar o sistema de rastreamento e alerta da população civil próxima.

3 ENGENHARIA DE SOFTWARE

A engenharia de software é uma vertente da engenharia que utiliza de métodos e processos para a o desenvolvimento e manutenção de aplicativos, sistemas e programas.

Atualmente, essas tecnologias e práticas englobam linguagens de programação, banco de dados, ferramentas, plataformas, bibliotecas, padrões de projeto de software, processo de software e qualidade de software. Além disso, a engenharia de software deve oferecer mecanismos para se planejar e gerenciar o processo de desenvolvimento de um sistema computacional de qualidade e que atenda às necessidades de um requisitante de software.

Os fundamentos científicos para a engenharia de software envolvem o uso de modelos abstratos e precisos que permitem ao engenheiro especificar, projetar, implementar e manter sistemas de software, avaliando e garantindo suas qualidades. (WIKIPEDIA).

3.1 Metodologias de desenvolvimento

O processo de desenvolvimento de software é um conjunto de atividades que almejam atingir as expectativas e necessidades de um usuário, cliente ou consumidor. Para isso, é utilizado diversos métodos e modelos, que devem ser adaptados aos moldes do projeto a ser implementado.

3.2 Metodologia Cascata (Waterfall)

A metodologia Cascata ou "Waterfall" é uma das principais metodologias tradicionais utilizadas no mercado de trabalho, também é utilizado como base de outras metodologias. Consiste em processos pré-definidos como a coleta das necessidades do cliente (definindo as regras de negócio), realização de diagramas (estrutura e design do sistema), codificação e aplicação das tecnologias e estruturas lógicas desenvolvidas, realização de testes e aperfeiçoamentos e fornecimento da devida manutenção. O cliente somente tem acesso ao produto final.

3.3 Metodologias tradicionais

Muitos modelos utilizam como base a metodologia Cascata, as metodologias "tradicionais" geralmente são utilizados quando há uma grande disponibilidade de tempo para o desenvolvimento do sistema, sem participação direta do cliente.

3.4 Metodologia Ágil

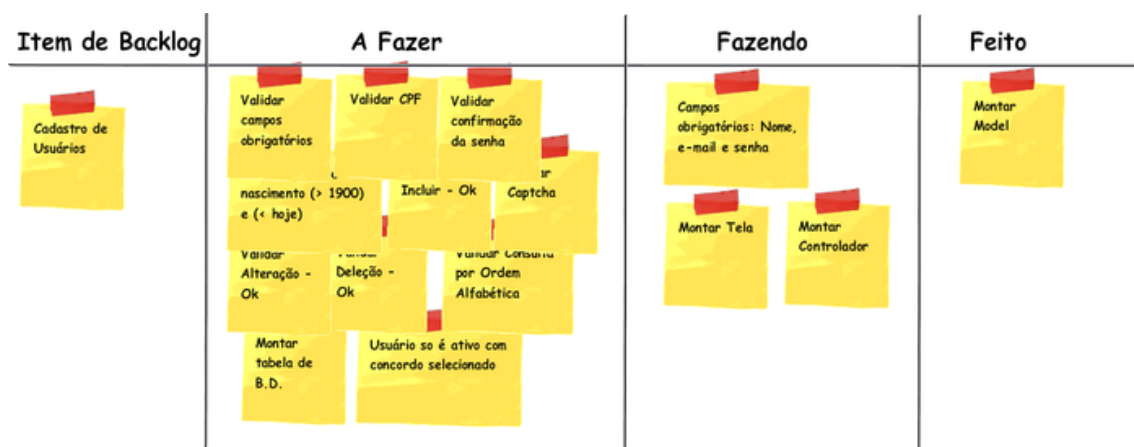
A metodologia Ágil trabalha com prazos menores e conta com a participação direta de seus clientes, possui alguns princípios como:

- Menor documentação
- Grande abertura para recebimento de alterações e feedbacks.
- Desenvolvedores, Product Owner e clientes trabalham juntos.
- Frequente realização de entregas parciais do software (Funções e resoluções de problemas do cliente) como medidas de progresso.

3.5 Quadro Kanban

O quadro Kanban é uma ferramenta de organização e administração de projetos, apresenta colunas de "a fazer", "fazendo" e "feito", adaptável para muitos tipos de negócios. Neste modelo, um ator é responsável por realizar as mudanças de status das atividades postadas no quadro. Tradicionalmente, o quadro é preenchido com post-its, mas hoje em dia, existem versões virtuais e online do quadro, muito utilizado quando parte da equipe envolvida atua remotamente.

Imagem 1 — Exemplo de uso do quadro Kanban



Fonte: <https://www.semeru.com.br/blog/wp-content/uploads/2012/09/teste.png>.

3.6 Utilização prática

Para realizar o projeto software, utilizamos os princípios das metodologias Ágeis. Com um prazo de desenvolvimento mais curto, as metodologias ágeis possuem mais flexibilidade, menor documentação e maior eficiência na aplicação de rotinas quando comparado a modelos "tradicionais".

Utilizamos o quadro Kankan para facilitar a visualização do projeto como um todo. Com uma análise de riscos, encontramos possíveis problemas como o mal uso do software e uma possível manipulação de dados de "má-fé". Para lidar com problemas como este, realizamos diagramas para auxiliar o desenvolvimento de rotinas que possam reduzir ou anular tais questões, dentre elas:

- Requisitos do sistema, definindo as rotinas e funções essenciais para a execução do software, como o sistema de cadastro, alteração e consulta de clientes, gerando de logs de usuário, que possibilita auditoria e rastreamento de más condutas.
- Rotinas de entrada de dados, polindo e garantido a integridade de dados essenciais (como CPF, CEP e data de nascimento).
- Rotinas de processamento de dados, para a transformação, ordenação, classificação e calculo, utilizando das funções e formulas criadas de forma particularizada.
- Rotinas de saída de dados, onde é o resultado da codificação é disponibilizado, exibindo os dados já formatados para o uso do profissional da saúde.
- Rotinas de padronização, alinhando e padronizando variáveis, utilizando comentários para facilitar a visualização e entendimento do código, ações essenciais para futuras manutenções.
- Rotina de testes, produzindo e executando diversos cenários a fim de reduzir eventuais "bugs".

4 LINGUAGEM DE PROGRAMAÇÃO C

Para o funcionamento de um software, o programador deve desenvolver a solução de uma problemática e aplicar essa lógica, na qual irá ser executada de forma literal pelo computador, executando processos estruturados que devem respeitar a devida sintaxe da linguagem de programação escolhida.

O computador segue as instruções desenvolvidas pelo programador como um passo a passo, de cima para baixo, realizando cálculos matemáticos e armazenando esses dados na memória, onde poderão ser consultados para novos cálculos ou exibição de um resultado. Um bom exemplo seria o uso de variáveis que podem assumir qualquer valor, porém são restritas a um tipo, uma "classe" de dados.

4.1 Tipos de variáveis

A linguagem C utiliza como base os seguintes tipos de dados:

Tabela 1 — Tipos de Dados

| Tipos de dados (C) | Definição | Espaço em memória |
|--------------------|---------------------------------------------|-------------------|
| Char | Caractere | 8 Bits |
| String | Cadeia de caracteres | Variável |
| Int | Números inteiros | 6 Bits |
| Float | Números Reais | 32 Bits |
| Double | Números Reais com maior alocação de memória | 64 Bits |
| Void | Vazio | 0 |

Fonte: Os autores (2022).

Quando queremos armazenar a idade de um paciente por exemplo, utilizamos o tipo "int", que suporta somente números inteiros, perfeito para esta finalidade.

Existem outros tipos de dados, que permitem o armazenamento de maiores espaços de memória, ou que possuem funções específicas como armazenar apenas números positivos. Na linguagem C, com esses tipos de variáveis é possível manipular e criar de novos tipos de dados, "variáveis personalizadas", conhecidas como "Structs".

4.2 Estruturas de dados

As "structs", são uma estrutura de dados que permite ser "personalizada",

suporta diversos tipos de dados e é composta por uma ou mais variáveis. Por exemplo, para desenvolver este software, precisamos armazenar os dados pessoais dos pacientes diagnosticados com o covid-19, para isto, criamos estruturas de dados que nos permitem, em uma única variável, receber os valores de nome, endereço, e-mail e os demais dados.

Imagem 2 — Exemplo de estrutura de dados, tipo "pessoa".

```
typedef struct pessoa{  
    char cpf[12];  
    char nome[100];  
    char email[50];  
    char telefone[15];  
    struct endereco moradia;  
    struct tm nascimento;  
    int idade;  
    char comorbidades[10][25];  
    bool grupo_de_risco;  
    struct log_log_usuario;  
}pessoa;
```

Fonte: Os autores (2022).

No caso acima, a estrutura de dados do tipo "pessoa" possui variáveis que irão receber e armazenar os dados fornecidos pelos profissionais da saúde, afunilando todos os dados em um único lugar, facilitando e organizando o código desenvolvido.

4.3 Laços de repetição

Os laços de repetição ou "loops" são estruturas que permitem a iteração de um código até que uma certa condição seja atingida. Na linguagem de programação C, existem 3 tipos de loops com diferentes finalidades.

4.3.1 Laço for

O loop "For" é muito utilizado na linguagem C quando sabemos com antecedência quantas vezes a repetição deverá ser executada, utilizando uma variável para controlar a contagem e incrementar seu valor, até que sua condição se torne verdadeira.

4.3.2 Laço While

O laço "while" é utilizado quando não sabemos quantas vezes uma ação deve

ser realizada até sua condição se tornar verdadeira. Caso sua condição seja inicialmente verdadeira, não executa seu bloco de ações.

4.3.3 Laço Do-While

O laço "do while" funciona de forma similar ao "while", porém, obrigatoriamente realiza o algoritmo (bloco de ações) pelo menos uma vez antes de checar sua condição de parada, realizando o loop quantas vezes forem necessárias. Este laço foi frequentemente utilizado para obter uma entrada de dados específica.

Imagem 3 — Exemplo de uso Do-while

```
do
{
    setbuf(stdin, 0);
    printf("\n Digite o CEP da casa: \n\n\tCEP: ");
    scanf("%s", &novo->paciente.moradia.cep);
    tamanho_string = strlen(novo->paciente.moradia.cep);
    checar_digito = checar_digitos(novo->paciente.moradia.cep);
}
while(!(tamanho_string == 8 && checar_digito == 1));
```

Fonte: Os autores (2022).

Para ter certeza de que a entrada de dados do usuário seja um número de CEP possivelmente válido, utilizamos um laço de repetição que solicita ao usuário um número de CEP e sua condição de parada é a obtenção de um conjunto numérico que seja constituído por oito algarismos.


4.4 Funções

As funções são algoritmos com uma finalidade definida, são trechos dos softwares que permitem serem invocados para a realização da tarefa ao qual foi projetada.

Seu significado e uso são muito parecidos com o de funções matemáticas, ou seja, existe um nome, uma definição e posterior invocação à função. Vamos explicitar o conceito de função matemática para depois verificar que a contra-parte em computação é semelhante. Tomemos um exemplo simples, função cúbica: $f(x) = x^2$ é sua definição (eventualmente explicitamos $f: \mathbb{R} \rightarrow \mathbb{R}$) e exemplos de seu uso poderia ser $f(-1)$ e $f(1/2)$, que devolvem, respectivamente, 1 e 1/4 (BRANDÃO).

Utilizamos funções quando precisamos realizar a mesma tarefa repetidas vezes, funções podem invocar outras funções como no exemplo:

Imagem 4 — Exemplo de função na linguagem C



```

char* converter_string_minusculo(char *string)
{
    int comprimento_string = strlen(string);
    for (int i = 0; i < comprimento_string; i++)
    {
        string[i] = tolower(string[i]);
    }
    return string;
}

```

Fonte: Os autores (2022).

A função denominada "converter_string_minusculo" recebe como argumento para a sua execução, uma string (cadeia de caracteres). Com isso, é invocada outra função "strlen" que realiza a contagem do número de caracteres presentes na string e armazena este número na variável do tipo inteiro "comprimento_string". Com estas informações é possível realizar um laço de repetição que irá acessar, de forma linear e singular, todos os caracteres de qualquer string fornecida, e em cada acesso, será invocada outra função "tolower" que irá transformar, se necessário, o caractere acessado em sua versão minúscula.

No desenvolvimento do software, a função "converter_string_minusculo" é chamada sempre que necessitamos de padronizar um dado fornecido pelo usuário. Um exemplo prático de sua funcionalidade, seria receber do usuário uma string de entrada com possibilidades ("Sim", "sim", "SIM", "slm", etc.) como argumento da função, e transforma-la em um padrão de letras minúsculas "sim".

4.5 Bibliotecas de funções

As bibliotecas são conjuntos de funções prontas que se só tornam disponíveis para uso após inseri-las no cabeçalho do programa.

Todos os programas em linguagem C usam funções das bibliotecas padrão da linguagem. O conjunto de funções de cada biblioteca é descrito em um arquivo-interface (= header-file), que tem o mesmo nome da biblioteca e sufixo .h. (Essa interface também é conhecida como API, ou application programming interface.) (FEOFILOFF, 2018).

Imagem 5 — Bibliotecas utilizadas para o desenvolvimento do Software

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <stdbool.h>
4  #include <string.h>
5  #include <ctype.h>
6  #include <time.h>
7  #include <conio.h>
8
```

Fonte: Os autores (2022).

Deste modo, o programador pode inserir bibliotecas no software conforme as necessidades, tornando-o mais rápido e leve.

4.6 Listas encadeadas

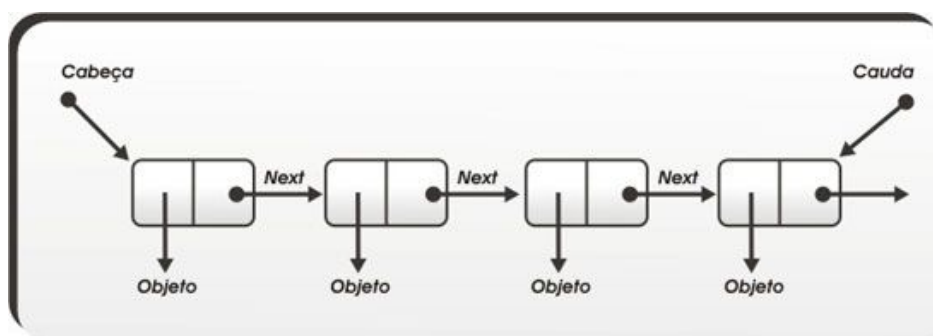
Uma lista encadeada ou "lista ligada" é uma estrutura de lista de objetos do mesmo tipo que estão interligadas por uma variável "nó" (uma variável do tipo ponteiro cujo único propósito é indicar a localização do próximo item da lista na memória do computador).

Em uma lista ou "cadeia" comum, os dados ficam armazenados em um mesmo espaço de memória e para isso é necessário declarar de antemão a capacidade máxima da lista. Ao declarar previamente o tamanho de uma lista surgem duas possibilidades negativas:

- Utilizar memória excessiva;
- Limitar a capacidade da armazenamento da lista.

Uma solução para o problema é a utilização de lista encadeada, para isso é necessário a alocação dinâmica de memória, através da função "malloc()" que reserva diferentes espaços de memória, porém continuam interligados pela variável "nó" que guarda a informação do próximo elemento da lista. Quando a variável "nó" se encontra com o valor "NULL" ou nulo, este se elemento torna o fim da lista.

Imagem 6 — Exemplo de funcionamento de uma lista encadeada



Fonte: <https://upload.wikimedia.org/wikipedia/commons/6/69/ListaEncadeada.jpg>.

Com este propósito, nosso software utiliza de listas encadeadas para armazenar os dados dos pacientes cadastrados, deste modo, a quantidade de pacientes cadastrados suportada é limitado somente pelo hardware da máquina em que o programa está sendo executado.

Outra preocupação, foi reduzir os passos de busca por elementos dentro desta lista, para esse fim, foi criada uma tabela Hash do tipo "nó", chamada de "indice_cpf[100]" onde cada cadastro será salvo em seu respectivo índice. Este índice é gerado pela função "obter_indice()" que retorna os dois últimos dígitos do CPF (o CPF dado foi escolhido por se tratar de um documento único e permanente). Um paciente com CPF final "51" terá seus dados salvos na lista encadeada de posição 51 ("indice_cpf[51]"), aumentando significativamente a velocidade de busca dos dados.

5 FUNCIONAMENTO DO SOFTWARE

Desenvolvemos um software capaz de suprir as necessidades informadas na problemática, o mesmo possui um conjunto de estruturas e funções que foi pensado exclusivamente para esta demanda.

5.1 Estruturas desenvolvidas

Para resolver problemas de armazenamento de variáveis, utilizamos da função "typedef struct" que define um tipo "personalizado" de dado, sendo eles:

- Estrutura de dados para o controle de acesso e função Login.

Imagem 7 — Estrutura tipo "login".

```
typedef struct login{  
    char usuario_login[25];  
    char usuario_senha[25];  
    char usuario_nome[100];  
    char usuario_cpf[12];  
}login;
```

Fonte: Os autores (2022).

- Estrutura de dados para receber e armazenar endereços:

Imagem 8 — Estrutura tipo "Endereço".

```
typedef struct endereco{  
    char cep[9];  
    char rua[50];  
    char numero[10];  
    char bairro[50];  
    char cidade[50];  
    char estado[3];  
}endereco;
```

Fonte: Os autores (2022).

- Estrutura de dados para o controle de log gerado pelo usuário.

Imagem 9 — Estrutura tipo "log".

```
typedef struct log{
    char cadastrado_por[100];
    struct tm data_cadastro;
    char alterado_por[100];
    struct tm data_alteracao;
    bool usuario_alterado;
}log;
```

Fonte: Os autores (2022).

- A estrutura de dados do tipo pessoa, é onde será armazenado todas as informações vinculadas a um paciente.

Imagem 10 — Estrutura tipo "pessoa".

```
typedef struct pessoa{
    char cpf[12];
    char nome[100];
    char email[50];
    char telefone[15];
    struct endereco moradia;
    struct tm nascimento;
    int idade;
    char comorbidades[10][25];
    bool grupo_de_risco;
    struct log log_usuario;
}pessoa;
```

Fonte: Os autores (2022).

- Estrutura desenvolvida para o funcionamento da lista encadeada:

Imagem 11 — Estrutura tipo "Nó".

```
typedef struct No{
    pessoa paciente;
    struct No *proximo;
}No;
```

Fonte: Os autores (2022).

5.2 Funções e Procedimentos

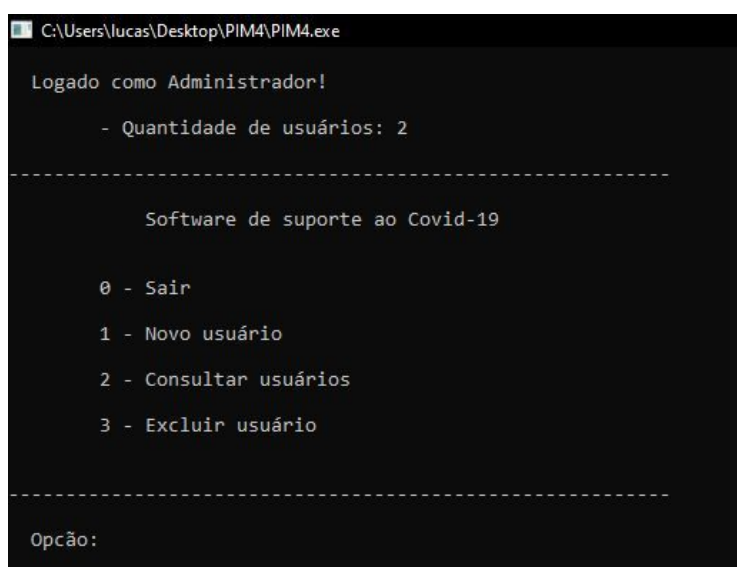
Ao ser inicializado, o programa irá executar a função "carregar_lista_usuarios()" para verificar se é a primeira execução do sistema ou se há um arquivo com dados de usuários para ser lido, em seguida, a função "login_senha()" realiza o primeiro contato entre o usuário e o software, solicitando os

dados de login (usuário e senha) para permitir o acesso ao sistema, neste momento, caso seja feito um acesso administrativo (padrão - usuário: ADMIN, senha: ADMIN), teremos o seguinte retorno:

Neste exemplo, temos 2 usuários cadastrados, sendo assim, o menu permite:

- 0 - Sair, retornando ao sistema de login.
- 1 - Cadastrar um novo usuário (usuário, senha, nome completo e CPF).
- 2 - Consultar usuários já cadastrados, caso exista.
- 3 - Excluir acesso, removendo o acesso e dados de um usuário previamente cadastrado.

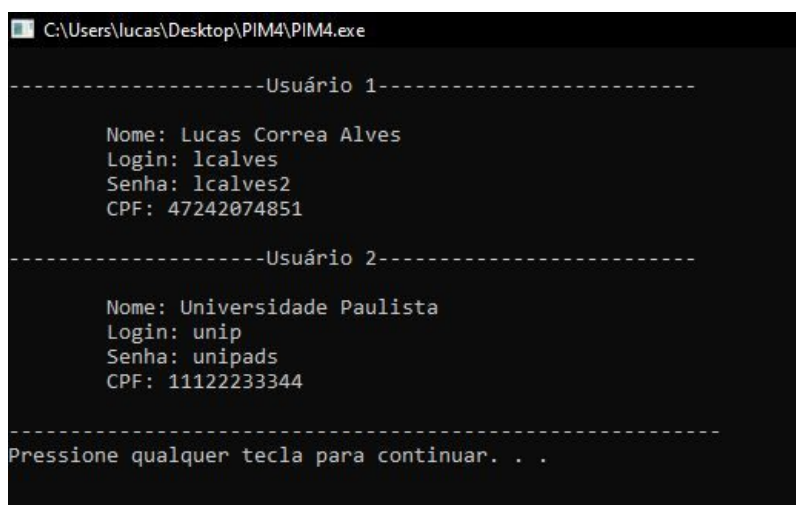
Imagem 12 — Acesso administrativo do sistema



Fonte: Os autores (2022).

Ao realizar a consulta de usuários, o programa irá fornecer as devidas informações:

Imagem 13 — Função de consulta de usuários



```
C:\Users\lucas\Desktop\PIM4\PIM4.exe

-----Usuário 1-----

Nome: Lucas Correa Alves
Login: lcalves
Senha: lcalves2
CPF: 47242074851

-----Usuário 2-----

Nome: Universidade Paulista
Login: unip
Senha: unipads
CPF: 11122233344

-----
Pressione qualquer tecla para continuar. . .
```

Fonte: Os autores (2022).

5.3 Menu principal

Após um login bem sucedido, o programa irá disponibilizar dados como a data atual, usuário logado, quantidade de pacientes qualificados no grupo de risco e o total de cadastros. Em seguida, exibe um menu com cinco opções, permitindo a realização de cadastros, alteração de pacientes previamente registrados e consultas.

Imagem 14 — Execução do menu principal do software



Fonte: Os autores (2022).

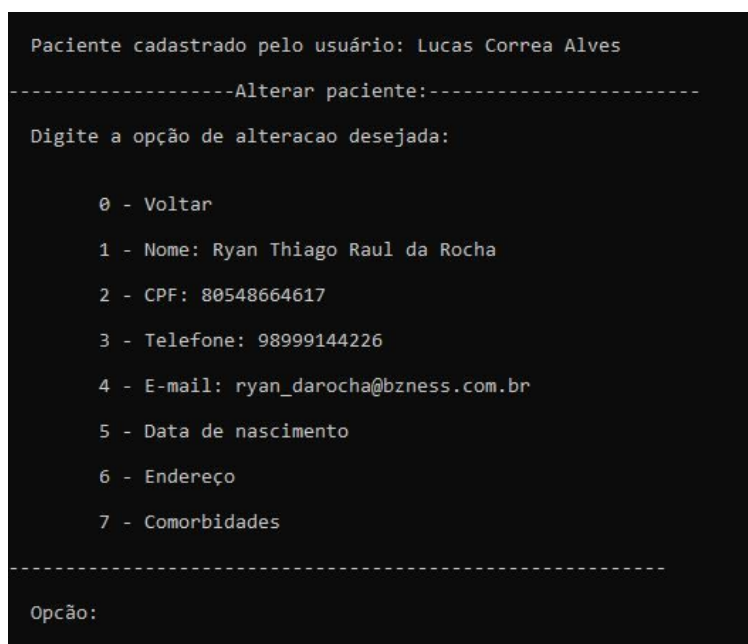
5.4 Cadastro de paciente

Esta é a principal função do software, nesta opção, o usuário deve fornecer ao sistema um CPF válido e não cadastrado, juntamente dos demais dados pessoais dos pacientes diagnosticados com o Covid-19. Após o cadastro, o sistema irá registrar o log do usuário na ficha do paciente (registrando o nome do usuário e data de cadastro) e irá salvar estes dados em um arquivo binário, que irá atuar como banco de dados para o programa.

5.5 Alterar dados cadastrados

Esta terceira opção permite que o usuário possa alterar todos os dados previamente cadastrados, fornecendo um CPF válido como entrada para a busca. Ao finalizar as alterações, o sistema registra o log de usuário na ficha cadastral do paciente e as salva no arquivo, substituindo os dados anteriormente registrados. Este log ficará disponível em uma próxima consulta, onde irá constar a data e o nome do usuário que realizou a última alteração na ficha do paciente, permitindo um maior controle sob as informações registradas e prevenindo alterações maliciosas.

Imagem 15 — Menu de alterações



Fonte: Os autores (2022).

5.6 Consultar dados do paciente

Esta opção permite que o usuário consiga visualizar os dados de um paciente cadastrado, são fornecidas todas as informações pessoais, lista de comorbidades (se houver), o usuário que efetuou o cadastro, o que realizou a última alteração (se houver) e status relacionado ao grupo de risco.

Imagem 16 — Exibição dos dados consultados de um paciente fictício durante a execução do software.

```
C:\Users\lucas\Desktop\PIM4\PIM4.exe

-----Paciente:-----

Nome: Marli Jennifer Alessandra Vieira
CPF: 58374374799
Telefone: (49)39422588
E-mail: marli.jennifer.vieira@tce.sp.gov.br
Idade: 49
Data de nascimento 01/10/1973
CEP: 88521515
Endereço: Rua Vito Pedro Boscatto
Número: 682
Bairro: Tributo
Cidade: Lages
Estado: SC.
Cadastrado pelo usuário: Lucas Correa Alves      Data: 26/11/2022.
Alterado pelo usuário: Universidade Paulista     Data: 26/11/2022.
Grupo de risco: Sim.

-----Lista de comorbidades:-----

1 - Obesidade

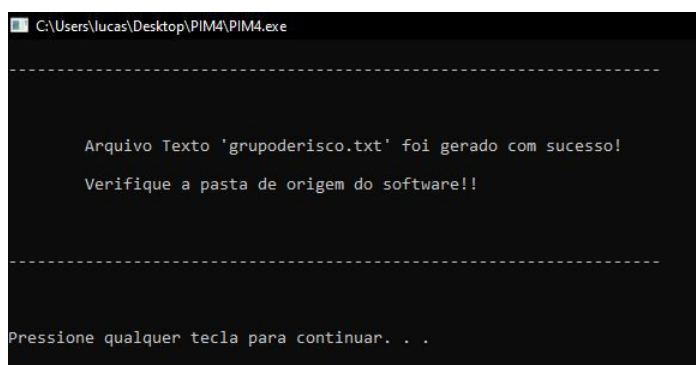
-----
Pressione qualquer tecla para continuar. . .
```

Fonte: Os autores (2022).

5.7 Arquivo Texto

Ao selecionar a opção número 4, o programa irá buscar as últimas informações registradas banco de dados em busca de pacientes qualificados no grupo de risco, ao identificar este paciente, o sistema registra em um arquivo texto "grupoderisco.txt" os dados de CPF e idade, que devem ser fornecidos como um relatório para a central da Secretaria da Saúde.

Imagem 17 — Arquivo "grupoderisco.txt" sendo gerado pelo software.



Fonte: Os autores (2022).

Se já houver um arquivo de mesmo nome na pasta de origem do sistema, o programa irá substituir o arquivo com as informações mais recentes.

Imagem 18 — Arquivo gerado pelo software, exemplo de relatório.



Fonte: Os autores (2022).

6 CONCLUSÃO

Em virtude dos fatos mencionados, entende-se que as medidas adotadas para o combate a pandemia trouxeram mudanças permanentes na sociedade em que conhecemos. Os novos hábitos e o avanços tecnológicos trouxeram consigo uma grande redução de custos (sobretudo em tempo e transporte), principalmente devido ao uso de softwares inteligentes e a alta disponibilidade e velocidade de internet, afetando setores como educação e trabalho a distancia, derrubando barreiras culturais e tradicionais onde tais práticas eram vistas com maus olhos.

Ao iniciar o processo de desenvolvimento, ficou claro que a utilização dos princípios da engenharia de software é um fator imprescindível para um bom desenvolvimento de projeto, adaptando as metodologias, e quando necessário, utilizar de análise de risco perante aos desafios, ambientes e demais variáveis incontroláveis.

As técnicas de programação, em união com a lógica aplicada, quando devidamente utilizada, se mostram uma ferramenta poderosa, capaz de reduzir custos de mão de obra e trazer praticidade ao usuário. A linguagem de programação C é uma linguagem flexível, de "baixo nível", que utiliza de funções e processamento de dados em seu funcionamento, voltada para uso geral, é leve demanda um maior nível técnico ao trabalhar com processos mais complexos.

Hoje, existem linguagens de programação de "alto nível", capazes de realizar tarefas como a automação de processos complexos, o uso de inteligência artificial e utilização de ferramentas multifuncionais conectadas a internet, o que permite maiores possibilidades de aplicação.

Visto isso, é possível concluir que ao adaptar os modelos de desenvolvimento, seja ágil ou tradicional, com uma boa aplicação técnica e tecnológica, o resultado será um software que cumpre seu principal propósito, o de resolver problemas.

REFERÊNCIAS

BRANDÃO, Leônidas. **Um paralelo entre função matemática e função em linguagens de programação**. 1 p. Disponível em: https://www.ime.usp.br/~leo/mac2166/2017-1/introducao_funcoes.html. Acesso em: 25 nov. 2022.

CASAVELLA, Eduardo. **Exibindo data e hora com time.h e localtime**. 1 p. Disponível em: <http://linguagemc.com.br/exibindo-data-e-hora-com-time-h/>. Acesso em: 10 nov. 2022.

FEOFILOFF, Paulo. **Bibliotecas de funções**. 2018. 1 p. Disponível em: <https://www.ime.usp.br/~pf/algoritmos/apend/interfaces.html>. Acesso em: 25 nov. 2022.

GASPAR, Wagner. **Lista encadeada, lista duplamente encadeada e lista circular**. 1 p. Disponível em: <https://wagnergaspar.com/lista-encadeada-lista-duplamente-encadeada-e-lista-circular/>. Acesso em: 12 nov. 2022.

WIKIPEDIA. **Engenharia de software**. 1 p. Disponível em: https://pt.wikipedia.org/wiki/Engenharia_de_software. Acesso em: 26 nov. 2022.