UNIVERSIDADE PAULISTA

ICET - INSTITUTO DE CIÊNCIAS EXATAS E TECNOLOGIA

CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

PROJETO INTEGRADO MULTIDISCIPLINAR

PIM II

SOFTWARE DE CONTROLE

Rede de Clinicas

Nome	R.A
Bianca Siqueira B. Sousa	N610751
Daniel Arruda Santos	F328JJ7
Daniel Rodrigues de Oliveira	N6105B1
David De Col Rodrigues	F211DDO
Rafael Nascimento Aguiar	N627913

SÃO PAULO – SP

DEZEMBRO/2020

NOMES	RA
Bianca Siqueira B. Sousa	N610751
Daniel Arruda Santos	F328JJ7
Daniel Rodrigues de Oliveira	N6105B1
David De Col Rodrigues	F211DD0
Rafael Nascimento Aguiar	N627913

SOFTWARE DE CONTROLE
Rede de Clinicas

Projeto Integrado Multidisciplinar (PIM) desenvolvido como exigência parcial dos requisitos obrigatórios à aprovação semestral no Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas da UNIP (Universidade Paulista), orientado pelo corpo docente do curso.

São Paulo – SP

Dezembro / 2020

RESUMO

Neste trabalho foi desenvolvido um sistema de controle para uma rede de clinicas medicas com três unidade localizadas em uma cidade no Brasil. Para desenvolvemos este projeto foi necessário primeiramente de uma rápido estudo e reunião com o cliente e nosso grupo para sabemos como eles queriam o sistema e os seus requisitos principais para podemos definir com metas e objetivos a serem alcançados pela nossa equipe.

Depois de sabemos as nossas metas dividimos a nosso grupo e começamos a nos preparar para o desenvolvimento do projeto, começamos com as telas que seriam usadas nos softwares e as principais funções, fizemos um estudo de infraestrutura de rede para entendemos melhor como iriamos implementamos isto nas redes de clínicas. Separamos 2 programadores que foram responsáveis pelo o código e o software em geral e os outros 3 foram responsáveis pela os estudos em geral, manual e documentação do software e pela implantação de infraestrutura de rede.

Depois de semanas de trabalho e esforços de toda a obtemos um bom resultado que agradou o cliente de forma geral batendo as metas e objetivos dentro do prazo estimado pelo nosso grupo e o cliente.

Concluindo o nosso trabalho acreditamos que recebemos ótimos conhecimento sobre a linguagem C (que foi utilizada para o software) e sobre metodologias e infraestrutura de redes que contribuirá para mais desenvolvimentos de sistemas deste mesmo grupo

Palavras-Chave: Infraestrutura, Software, Clínica, Metas, Objetivos

Sumário

1. INTRODUÇÃO

O projeto PIM 2º semestre irá conter diversos temas que foram utilizados como base para aplicar em nosso trabalho, tendo como o objetivo de construir um sistema de atendimento ao cliente em uma rede de clínicas. Sendo assim contendo Linguagem em C que consiste em falar sobre sua definição, características, desenvolvimento e sua importância no trabalho para desenvolvermos na nossa pequena rede de clínicas. Trabalhamos também com Engenharia de Software 1 que diz sobre seu surgimento, sua importância em épocas passadas e que foi se desenvolvendo até os dias de hoje para uns melhores gerenciamentos de projetos que envolve um conjunto de atividades que faz um programa ser confiável e eficaz.

Outro tema que utilizamos foi Fundamentos de redes de dados e comunicação que contém sua definição e as explicações dos tipos de rede mais populares. Disserta-se também sobre Matemática para computação que fala sua definição, quando surgiu sua importância no mercado de trabalho. E por último, mas não menos importante temos a Ética e legislação profissional que explica termos ética como profissionais na área de trabalho e a importância de termos conhecimento sobre legislação que regulamenta as atividades profissionais.

2. LINGUAGEM E TÉCNICAS DE PROGRAMAÇÃO

2.1. Sobre Linguagem C

A linguagem C foi desenvolvida em 1972 por Dennis Ritche para ser utilizada com o sistema operacional UNIX. A linguagem teve grande aceitação e uso, sendo utilizada por diversos programadores nos dias de hoje.

Uma de suas qualidades é a flexibilidade que ela oferece ao programador. Existem diversas vantagens em utilizar a linguagem C, entre elas: possui um conjunto compacto de palavras-chaves e tipos de dados, evitando assim diversas operações desnecessárias e contém uso de ponteiros que permitem o acesso de baixo nível a memória.

A linguagem C é considerada de baixo nível ou de nível médio, o que possibilita um melhor controle do hardware, podendo manipular bits, bytes e endereços. Devido ao grande número de programadores que utilizam essa linguagem, existe uma vasta gama de compiladores e bibliotecas disponíveis no mercado, sendo alguns gratuitos e outros pagos.

Para que todos esses compiladores possam ser compatíveis entre si, existe o padrão C que foi estabelecido pelo comitê da ANSI (American National Standards Institute) em 1983.

2.3. Por Que Utilizar C Em Terminal

Por ser uma linguagem de baixo nível, com uma sintaxe semelhante à linguagem humana e um conjunto de funcionalidades enxuto, a linguagem C oferece um ótimo custo-benefício para projetos com baixo investimento e para usuários que estão começando a programar ou ainda estão entrando no mundo da tecnologia. C é uma linguagem de programação que precisa ser compilada, isto é, ela precisa ser transformada em um conjunto de códigos que a máquina interpretará para, então, criar um executável. Para fazer isso, a linguagem precisa de um compilador, que é um software que faz esse processado, a linguagem é considerada auto hospedada, pois com ela é possível criar o próprio compilador. Essa característica também é conhecida como linguagem completa.

2.4. Glossário Linguagem C

I/O – Denomina-se I/O (Input / Output) sistemas que fazem uso intensivo de entrada e saída de dados. Dentro da área de computação, a terminação I/O também é usada muito para expressar tecnologias novas de implementação**.**

Compilador – O compilador é um programa que transforma o seu código em algo que possa ser interpretado pela máquina. Os micros controladores compreendem apenas byte Codes, ou outras linguagens de baixo nível indicadas em seus firmwares, de modo que se faz necessário converter o seu código para algo que a máquina entenda.

Variável – Trata-se de um lugar reservado na memória para poder armazenar um dado, como um número, um caractere, um digito, entre outros tipos de dados.

Função – É um trecho de código com o propósito de executar um procedimento e, geralmente, retornar uma informação.

2.5. Desenvolvimento

Para os processos de codificação, gerenciamento, implementação e execução do sistema, será utilizado o IDE (Integrated Development Enviroment) Code:Blocks. O uso de uma IDE auxilia nos processos de escrita, manutenção e correção do programa, e outro grande benefício das IDEs consiste em já possuir um compilador integrado, tornando mais fácil o processo de compilar e corrigir erros. Quando escrevemos um programa em alguma linguagem de programação, o computador não entende diretamente o código que escrevemos, é necessário fazer um processo de tradução, traduzindo o nosso código para código de máquina, os zeros e uns que as máquinas entendem. O compilador é o software (programa de computador) responsável por essa tradução.

3. ENGENHARIA DE SOFTWARE I

Para os processos de codificação, gerenciamento, implementação e execução do sistema, será utilizado o IDE (Integrated Development Enviroment) Code:Blocks. O uso de uma IDE auxilia nos processos de escrita, manutenção e correção do programa, e outro grande benefício das IDEs consiste em já possuir um compilador integrado, tornando mais fácil o processo de compilar e corrigir erros. Quando escrevemos um programa em alguma linguagem de programação, o computador não entende diretamente o código que escrevemos, é necessário fazer um processo de tradução, traduzindo o nosso código para código de máquina, os zeros e uns que as máquinas entendem. O compilador é o software (programa de computador) responsável por essa tradução.

3.1Engenharia De Software

O termo “engenharia de software” apareceu pela primeira vez no ano de 1968, após a crise do software, época onde ocorriam dificuldades no desenvolvimento de programas livres de defeitos, confiáveis e eficientes.

Um software pode ser criado para atender as necessidades de um cliente, empresa ou para uso pessoal, com as técnicas que englobam linguagens de programação, base de dados, ferramentas, plataformas, padrões, processos e a qualidade de software.

Desenvolver um software pode ser um processo bastante complexo, exigindo uma equipe de trabalho disciplina, o gerenciamento de projetos e muitos recursos. Gerenciar projetos de software envolve um conjunto de atividades que são administradas de acordo com os parâmetros de custo, tempo e qualidade. Ao longo do processo de desenvolvimento de um software, devem ser utilizadas métricas quantitativas e qualitativas para que o produto final esteja de acordo com a necessidade e exigência do cliente.

No desenvolvimento de um software podem surgir alguns problemas, tais como o estouro de prazos e custos, a baixa qualidade devido ao excesso de erros, as mudanças próximas à data de entrega do produto, entre outros. Os profissionais da engenharia de

software são responsáveis por trabalhar nos quesitos e evitar os problemas que possam aparecer durante o desenvolvimento de um produto.

3.2. Aplicando Engenharia De Software O Projeto

Ao aplicar os conceitos de Engenharia de Software nos projetos é possível desenvolver um produto de alta qualidade, de forma eficiente e com um baixo custo, garantindo mais controle no processo de desenvolvimento e oferecendo um software que satisfaça os requisitos pré-estabelecidos.

Para este projeto foi adotado a modelo cascata de desenvolvimento de software, que consiste em dividir o ciclo de vida do projeto em quatro fases: análise, projeto, implementação e testes. Esse modelo é sistemático e sequencial, onde cada fase é estruturada como um conjunto de atividades que podem ser executadas por pessoas diferentes, simultaneamente. O que faz desse modelo o mais adequado, uma vez que os requisitos do software estão muito bem entendidos.

4. FUNDAMENTOS DE REDES DE DADOS E COMUNICAÇÃO

4.1.O Que São Redes De Computadores

As redes de computadores podem ser definidas como um conjunto de equipamentos que compartilham os mesmos recursos e trocam informações entre si. Um desses recursos é a conexão com a Internet que é dividida em todas as máquinas conectadas a uma determinada rede. Além disso, ela também possibilita a comunicação entre usuários, compartilhamento de informações e equipamentos.

Existem diversos tipos de redes, mas todas elas são definidas por dois fatores principais, que são o modelo de equipamento que serão conectadas a ela e distância que esses equipamentos se encontram um do outro.

4.2. Tipos de Redes mais populares:

Redes de área local (LAN): A LAN é uma junção de dispositivos dentro de uma determinada área. Cada dispositivo é representado como um “nó” na rede e ficam conectados ao servidor. Esse tipo de rede geralmente cobre uma pequena área e são interligadas através de tecnologia sem fio. A função principal de uma rede local é realizar a conexão de dispositivos para melhorar a produtividade e eficiência minimizando custos.

Redes de área pessoal (PAN): Essa rede é responsável por conectar aparelhos de uso pessoal (como computadores, celulares e tablets) por meio de uma conexão sem fio em uma área de até 10 metros. Ela permite que vários dispositivos se comuniquem internamente, focando o uso de apenas uma pessoa, o que a torna diferente da LAN. Se essa conexão for realizada via Bluetooth, é possível que até oito dispositivos se conectem a um aparelho principal (também conhecido como dispositivo mestre).

Wide área Network (WAN): Essa rede tem como função proporcionar uma conexão em uma distância maior. Na maioria das vezes essa rede vai servir como ponte de comunicação para diferentes LANs. Ela também pode ser utilizada para fazer a conexão de redes metropolitanas (MANs). A WAN também pode ser ligada a outras redes através de linhas de comunicação sem fio, mas essa conexão também pode ser feita através de cabos.

Metropolitan área netoworks (MAN): Essa rede é similar a uma rede local (LAN), mas consegue abranger uma cidade inteira. É formada pela conexão de várias LANs, unindo-as com linhas de backbone. Em uma rede metropolitana, diferentes LANs são conectadas através de uma central telefônica local. Alguns dos protocolos amplamente utilizados para a MAN são RS-232, X.25, Frame Relay, Asynchronous Transfer Mode (ATM), ISDN (Integrated Services Digital Network), ADSL (Asymmetrical Digital Subscriber Line), etc., no entanto, esses protocolos são bem diferentes daqueles usados para as LANs.

Apesar de estar em uma escala muito maior do que uma LAN, não cobre uma área tão grande quanto a WAN.

5. MATEMÁTICA PARA COMPUTAÇÃO

O curso de em Matemática Computacional foi implantado em 1999 o teve os seus primeiros egressos em 2002. Foi estruturado com o envolvimento dos cursos de Matemática, Ciência da Computação, Física e Estatística. Visa dar uma formação sólida em matemática e computação para profissionais que pretendam atuar na área de modelagem matemática e simulação e desenvolvimento de algoritmos. Na modelagem matemática estudam-se fenômenos naturais ou sociais por meio de conceitos matemáticos e da realização de simulações em computador.

A matemática computacional é uma área da matemática e da computação que trata do desenvolvimento de modelos matemáticos, para o tratamento de problemas complexos, e desenvolvimento de métodos numéricos de obtenção de soluções. Matemática computacional geralmente utiliza técnicas para solução numérica (aproximada) de problemas.

Esse profissional pode atuar em qualquer área que necessite de conhecimentos aprofundados das duas áreas: matemática e computação. Desta forma, é possível trabalhar no sistema financeiro e empresas de grande porte, que demandam logística mais apurada. Nesses locais, eles poderão resolver questões relacionadas à logística, economia e análise de riscos de investimento.

O matemático computacional também é encontrado em Instituições de pesquisa, nas áreas de Ciências Exatas, Biológicas e da Terra e em empresas geradoras de tecnologia. No ensino superior, são muitas as opções de emprego. “Temos vários concursos acontecendo para professor. Outra opção de trabalho é atuar em empresas de consultoria em matemática computacional.

6. ÉTICA E LEGISLAÇÃO PROFISSIONAL

É de extrema importância que além de conhecimentos técnicos, o profissional de TI seja dotado de competências humanas para que saiba tratar todos com respeito, dignidade, honestidade e cordialidade.

É essencial analisar a conduta ética dos mesmos que tem como obrigação zelar pela privacidade e proteção das informações, sejam elas organizacionais ou do cliente. Por tanto, é fundamental ter conhecimento da legislação que regulamenta as atividades profissionais da empresa e sobretudo, ter bom senso ao utilizar o recurso da mesma.

Os profissionais de TI são os principais profissionais que as empresas têm problemas com a falta de ética, por estarem trabalhando o tempo todo com essas informações, eles têm acessos ilimitados. É um problema que está acontecendo quase todos os dias, podemos verificar com frequência nos jornais, cracker invadindo empresas e roubando informações sigilosas e as vendendo para empresas concorrentes, entre outras características que configuram a falta de ética e até mesmo atos criminosos.

Atualmente, não temos um código de ética que seja direcionado especificamente aos profissionais da área de Tecnologia da Informação, apenas um projeto de lei que tramita no Congresso Nacional aguardando aprovação. Sendo assim, o gerente se torna responsável por estabelecer uma política interna que abranja o comportamento desses profissionais e que seja pautada na missão, visão e valores da empresa! Também é responsabilidade dele que essa política seja cumprida.

O comportamento ético do profissional de TI ao lidar com informações, e ao mesmo tempo, ter em mente o direito à privacidade do ser humano será o diferencial que conduzirá a sua conduta.

É de suma importância que questões éticas sejam identificadas e que haja conhecimento mais aprofundado sobre o comportamento humano e seus desdobramentos, para que assim, seja possível encontrar soluções que atendam estes problemas com eficácia.

As mudanças no ramo da tecnologia também acarretam modificações no comportamento dos indivíduos e cabe aos gerentes estabelecer políticas e conscientizar os profissionais da importância e da grande responsabilidade que possuem para o bom funcionamento da organização.

7. DESENVOLVIMENTO DO PROJETO

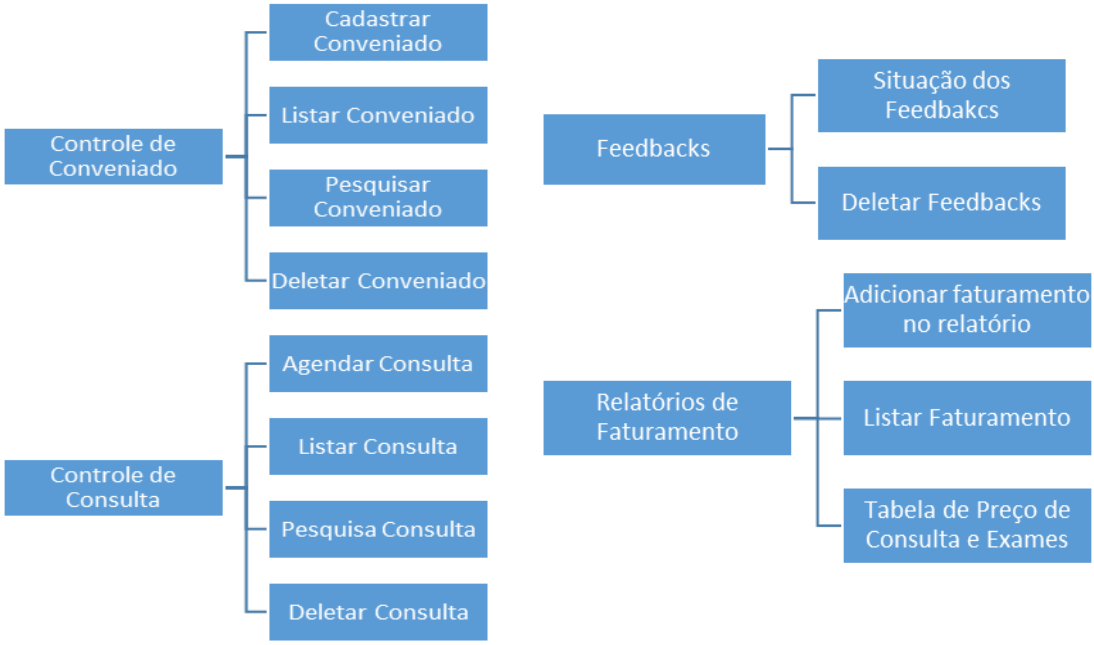
Uma pequena rede de serviços com apenas três unidades. Estas unidades estão próximas entre si, e localizadas no centro de uma cidade do Brasil. A rede da empresa necessita desenvolver um sistema para o gerenciamento e controle do negócio focado no atendimento aos clientes. Como a empresa não possui capital para grandes investimentos em infraestrutura, bem como softwares caros, optou por desenvolver o sistema na linguagem C, em modo console, armazenando os dados em arquivos texto com os diários para contabilização.

Cada unidade da rede possui pelo menos um computador no qual é executado o programa para atendimento aos clientes a ser desenvolvido. As três unidades da rede estão interligadas pela internet, formando uma intranet e uma extranet. Em uma das unidades há uma máquina adicional, para a qual os arquivos de fechamento diário das operações são copiados ao final do expediente.

Para suprir as necessidades pedidas pelo cliente desenvolvemos um sistema de gerenciamento e controle focado no atendimento aos clientes, com relatório mensal da clínica, cadastramento de conveniado, cadastramento de agendamento, relatórios cadastramento de feedbacks e tabela de preço dos exames e consultas.

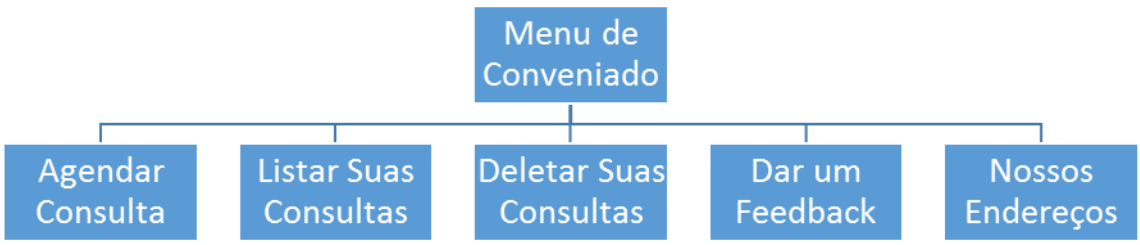
Com esse sistema criado os funcionários das clinicas junto com a equipe médica poderá ter um controle de conveniado podendo cadastrar, listar, pesquisar e deletar se preciso um conveniado e essas mesmas funções também servirão para o controle de agendamento de consultas.

A Figura **1** apresenta as opções no menu de atendimento que também apresenta algumas funções no menu médico



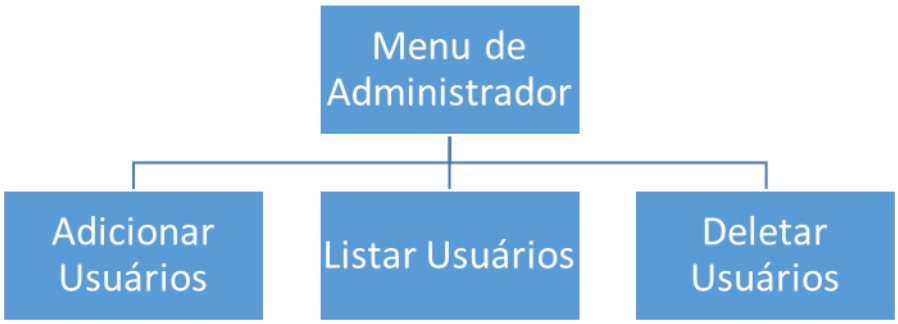
Esse sistema poderá ser utilizado também em um modo para conveniado para que o mesmo poder consultar suas consultas e exames, agendar consultas, desmarcar suas consultas, dar feedbacks para poder ajudar a clínica a descobrir o que o cliente achou da consulta, e ver os endereços das clínicas para saber onde encontrá-las.

A Figura **2** apresenta as opções no menu de conveniado



Caso seja preciso adicionar médicos e funcionários de atendimento ou excluir médicos e funcionários de atendimento no sistema ele já vem com uma conta administradora padrão criada para poder acessar e ter o controle dos usuários que utilizarão este sistema.

A Figura **3** apresenta as opções no menu de administrador



7.1 Caracterização Do Ambiente De Estudo

Em função do ambiente de estudo e conforme pesquisado identificamos que o mercado clinico em si passa por uns pós revolução segundo a “Regina Relva Romano” assessora do Ministério da Ciência, Tecnologia e Inovação (MCTI) e professora de inovação e tecnologia. Ela cita como exemplos a mudança repentina e as transformações não só nos setores médico, mas como em todos os setores da indústria mundial.

Em reunião com a diretoria das clinicas recebemos seus endereços e para podemos também ter uma base da região que ela atende para podemos nos preparamos melhor no desenvolvimento do sistema.

As clinicas não tem um número muito grande de pacientes mensais na casa dos 100 a 200 em alguns meses e com o numero variável de 3 a 6 com médicos por unidade de clínica com variáveis especialidades, assim com estas informações poderíamos ter uma ideia bem melhor de como funcionaria o sistema.

Enxergamos a tecnologia como a capacidade de aumentar o potencial humano. Na saúde, ela agrega facilidade, acessibilidade e informação, mas exige um bom planejamento de ação para que seja capaz de acolher o paciente por estes e outros motivos acrescentamos uma parte de conveniado para o mesmo poder acessar como um totem e ele precisar ir em umas das clínicas.

7.2 Desenvolvimento Do Sistema

O desenvolvimento do programa se dividiu por tela, e funções, Já na tela de login temos 3 opções de escolhas, “Sair, conveniados e Usuários”.

Se a opção for conveniada, irá ser levado a tela de Escolhas, onde temos as seguintes opções: “Agendar consulta, Listar Consultas, deletar suas Consultas, Dar Feedback e Nossos endereços.

- Opção agendar consulta, irá ser preenchido um formulário com;
- Primeiro nome
- Ultimo Nome
- Telefone
- Nome do Médico
- 9 (Nove) primeiros dígitos do CPF
- Dia, Mês e ano da consulta
- Horário da consulta
- Especialidade da consulta
- Número da carteirinha
- Nome da Unidade

Opção listar consulta, irá solicitar o número da carteirinha, se for **TRUE**, mostrará a lista. Opção deletar suas consultas, ir aparecer para digitar as seguintes informações:

- Nome
- Número da carteirinha
- Data da consulta
- Horário da consulta

Opção Dar Feedback, irá pedir para digitar o nome do médico.

Opção nossos endereços, irá aparecer todos os Endereços de nossas unidades

Caso sua escolha seja Usuários, uma verificação de Login e senha irá aparecer na tela para que você possa confirmar os dados do sistema, caso acerte o Login ele te levará para as respectivas telas de menu (Médico, Atendimento e Administrador) conforme a permissão cadastrada no login que foi acessado, tendo cada menu suas variedades e funções diferentes para poder atender as necessidades do cliente.

8. CONCLUSÃO

Sistema construída para que tenhamos facilidade e praticidade para transitar nos ambientes das redes hospitalares, de forma simples e direta, sistema foi construído para que pessoas familiarizadas ou não com o sistema, possa percorrer todas as funções sem maiores dificuldades e de forma intuitiva.

Se seu papel é como paciente a opções especiais para cada tipo de informação que deseja buscar no nosso sistema, seja realizar uma consulta, cancelar um agendamento de consulta a apenas saber onde se encontra nossas redes presenciais caso deseje um atendimentos e agendamento presencial, ou apenas saber onde se encontras nossas redes.

você medico ou administrador da rede, tem acesso privado para consultas de pacientes e listas de agendamentos, exclusões de pacientes de modo rápido, inserindo Seu e mail e senha da rede, uma rede privada irá se estabelecer, a onde apenas outros administradores tem acesso ao mesmo local, para facilitar e separar o acesso comum de pacientes de gestores, cada um conta com sua própria tela, com opções únicas para que a interatividade seja direta sem desvios de caminhos, facilitando e minimizando o tempo gasto na interatividade do sistema.

9. REFERÊNCIAS

CARVALHO, Ana. **A importância da ética profissional no segmento de tecnologia da informação**. Administradores.com café com adm. 09, 2010. Disponível em: <https://administradores.com.br/artigos/a-importancia-da-etica-profissional-no-segmento-de-tecnologia-da-informacao>. Acesso em: 26 Nov. 2020.

FLORENCE, Eduardo. **Ética para profissionais de ti**. Disponível em: < http://blogdoscursos.com.br/etica-para-profissionais-de-ti-4/> Acesso em: 26 Nov. 2020.

MELO, Alessandra. **Quais são os fundamentos de redes de computadores.** Disponível em: < https://ead.catolica.edu.br/blog/fundamentos-redes-de-computadores> Acesso em: 26 Nov. 2020.

MAYA, Alcides. **O que são redes de Computadores?** Alcides Maya. Disponível em: < https://alcidesmaya.edu.br/blog/182-o-que-sao-redes-de-computadores> Acesso em: 26 Nov. 2020.

ANÔNIMOS. **Lan**. Disponível em: < https://www.speedcheck.org/pt/wiki/lan/> Acesso em: 26 Nov. 2020.

ANÔNIMOS. **Conheça a rede de comunicação PAN – Personal Are Network**. Disponível em: < https://tndbrasil.com.br/conheca-a-pan-personal-area-network/> Acesso em: 26 Nov. 2020.

ANÔNIMOS. **O que é MAN (Metropolitan Area Network)**. Disponível em:< https://techenter.com.br/o-que-e-man-metropolitan-area-network/> Acesso em: 26 Nov. 2020.

PINHEIRO, José. **GAN - Global Area Network.** Disponível em:< https://www.projetoderedes.com.br/artigos/artigo_gan_global_area_network.php> Acesso em: 26 Nov. 2020.

ANÔNIMOS. **Redes de computadores; o que são e quais os principais tipos?** Disponível em:< https://netsupport.com.br/blog/redes-de-computadores/> Acesso em: 26 Nov. 2020.

GASPAR, Wagner. **Code: Blocks. O que é e como utilizar**. Disponível em:< https://wagnergaspar.com/codeblocks-o-que-e-e-como-utilizar/> Acesso em: 15 Nov. 2020.

MELO, Tiago. **Engenharia de Software: conceitos e aplicações**. Disponível em:< http://www.tiagodemelo.info/aulas/cefet/2007/aula-engenharia-software.pdf> Acesso em: 15 Nov. 2020.

ANÔNIMOS. **Questões sobre Ciclo de Vida de Software.** Disponível em:< https://www.mapadaprova.com.br/questoes/de/tecnologia-da-informacao/engenharia-de-software/ciclo-de-vida-de-software> Acesso em: 15 Nov. 2020.

YEGULALP, Serdar. **Por que a linguagem de programação C ainda está em alta?** Disponível em:< https://cio.com.br/carreira/por-que-a-linguagem-de-programacao-c-ainda-esta-em-alta> Acesso em: 15 Nov. 2020.

10.APÊNDICE

10.1. APÊNDICE A - MANUAL DE INSTALAÇÂO DO SOFTWARE

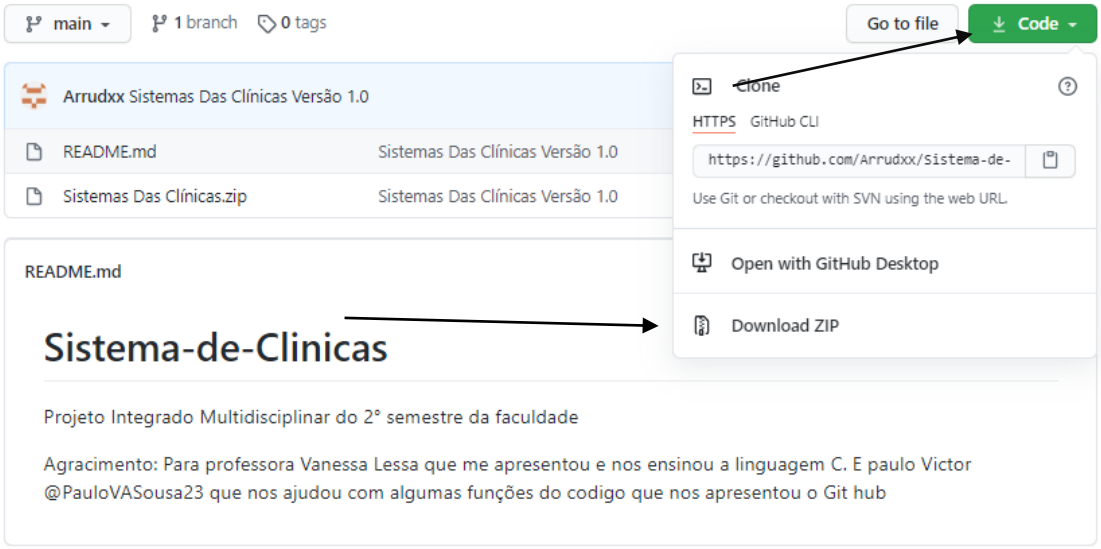
10.1.2. Requisitos Do Sistema:

- Sistema operacional: Windows® 7 Windows® 10
- Memória: 1 GB ou superior
- Processador: De 32 bits (x86) ou 64 bits (x64) de 1 GHz ou superior com vídeo integrado

10.1.3. Instalação do sistema para as clínicas:

Inicie o computador, abra o navegador de sua preferência e com o navegador aberto basta entrar no link.: https://github.com/Arrudxx/Sistema-de-Clinicas, apertar a aba code, clicar em Download Zip e esperar seu download começar

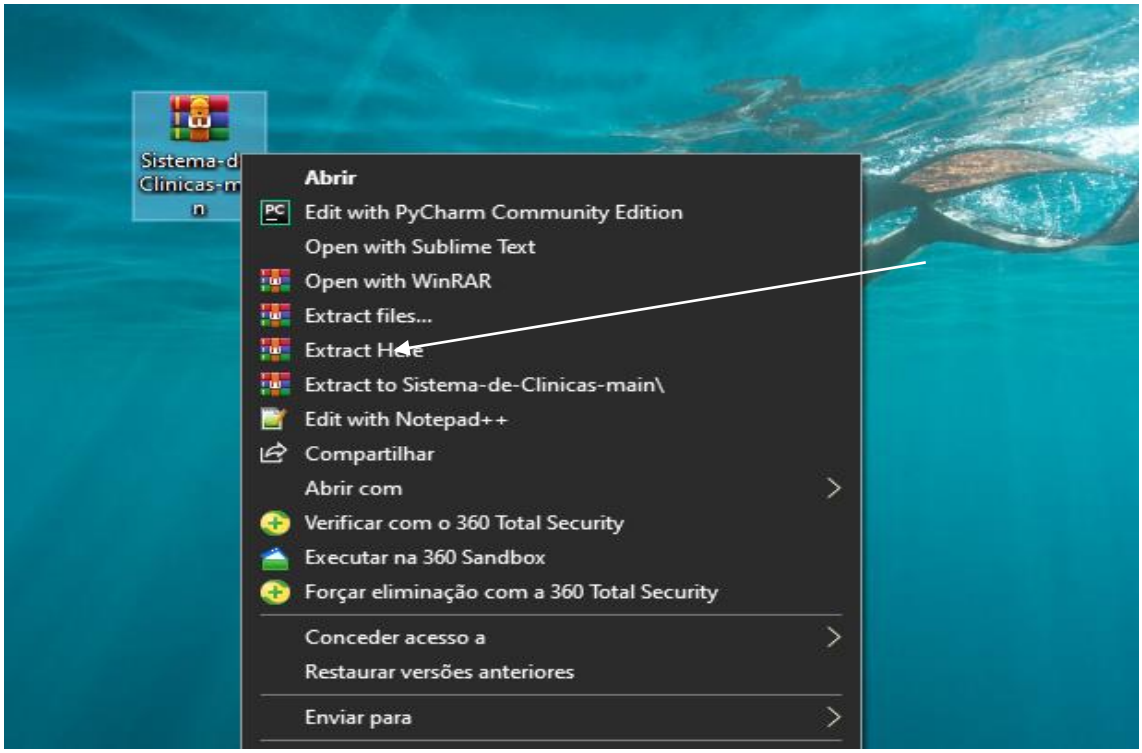
Figura **5** mostra em uma página de internet como fazer o download do sistema



Obs.: Para descompactar o arquivo sugerimos que baixem algum programa que abra arquivos .zip

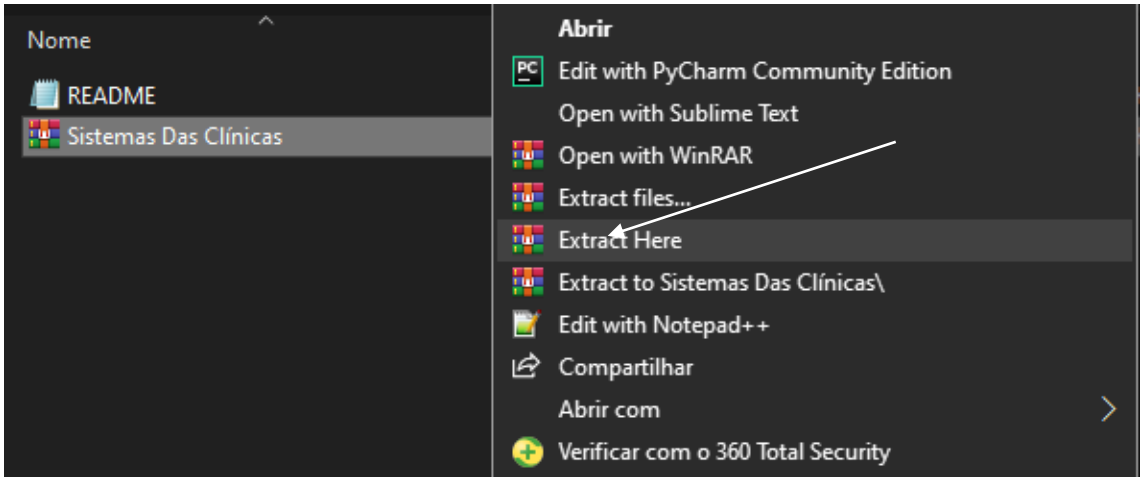
Com o arquivo .zip já no seu computador extrair o arquivo aqui ou na pasta de sua preferência

Figura **6** ensina como extrair os arquivos do sistema de um .zip



Entre na no arquivo que foi descompactado e também descompacte no mesmo local à única pasta chamada “Sistemas Das Clínicas”

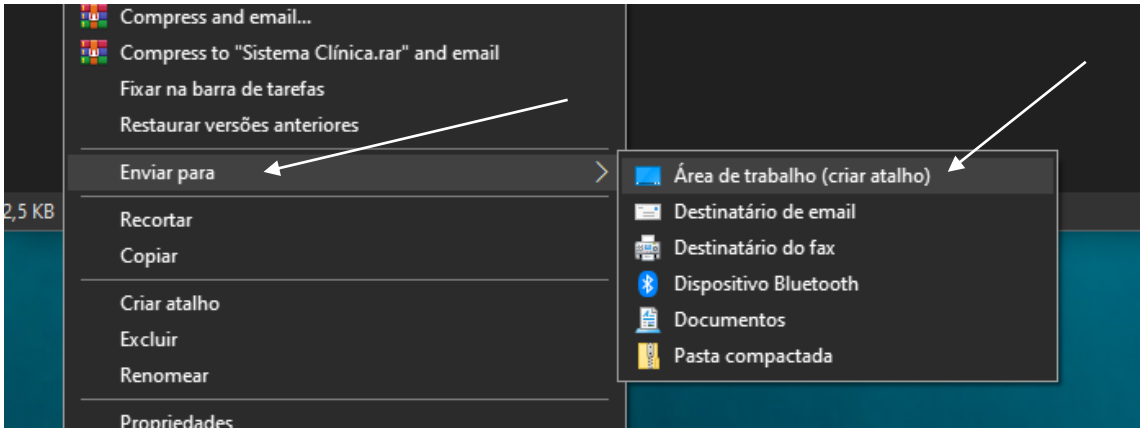
Figura **7** ensina como extrair a pasta “Sistemas Das Clínicas” de um .zip



Pronto agora é só entrar na pasta .Config

Caso queria deixar o sistema em um melhor lugar para ser usado basta puxar o arquivo “Sistema Clínica” para o lugar de preferência ou também aperte o botão direito do mouse, vá em “Enviar para” e clicar em “Área de trabalho (criar atalho) ”

Figura **8** ensina como criar um atalho do arquivo executável para a área de trabalho



10.2. APÊNDICE B - MANUAL DO SOFTWARE

10.2.1. Primeiro Acesso

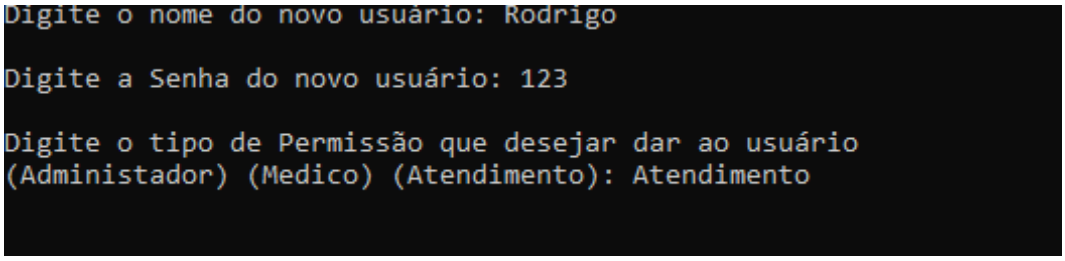
O primeiro passo para entrar no sistema é digitar o “2” para entrar em login dos usuários, depois aparecerá na tela para digitar o login e a senha. Por padrão, o sistema já vem com uma conta de administrador. Digite “Administrador” em login e “123” em senha, para acessar o menu de administrador e poder criar usuários! Para os demais acessos como atendente, médico e conveniado digite o respectivo login e senha.

Figura **9** Apresenta a tela de login do sistema

10.2.2. Adicionando Novos Usuários

No menu inicial digite a opção 1 “Adicionar Usuários”, em seguida digite o nome, cadastre a senha e conceda a permissão desejada, por fim, tecle “Enter” para salvar.

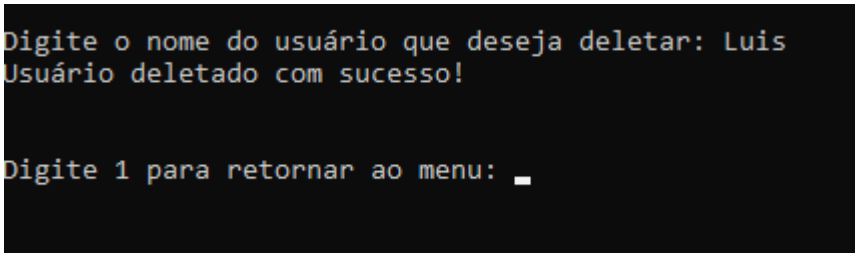
Figura **10** Mostra a tela de cadastramento de funcionário



10.2.3. Deletar usuário já existente

Para deletar um usuário selecione a opção 3 “Deletar Usuário” no menu inicial, em seguida digite o nome do usuário a ser deletado e tecle “Enter”.

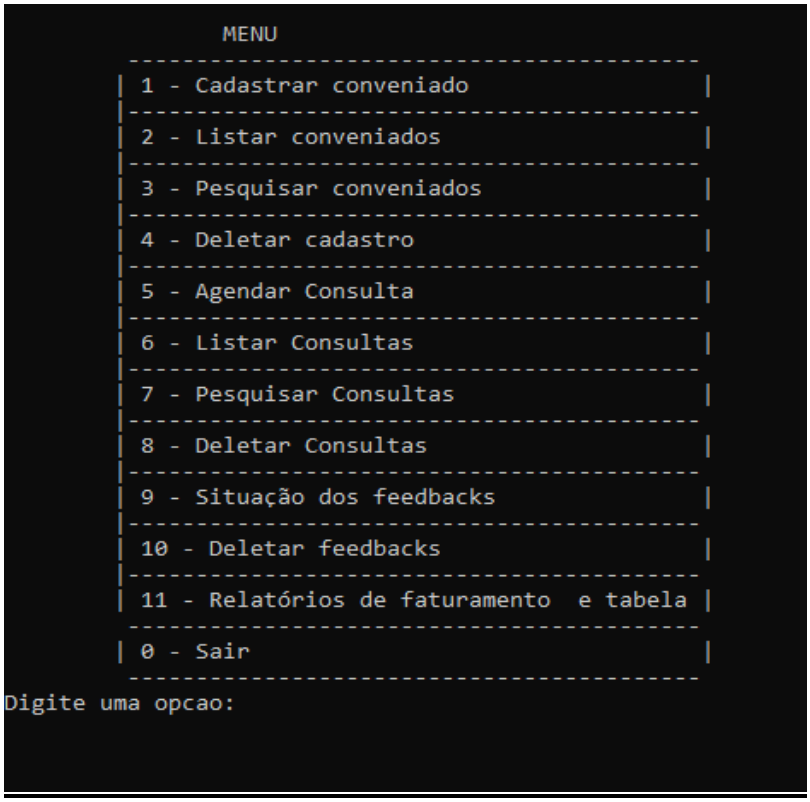
Figura **11** exibe que o usuário foi deletado com sucesso



10.2.4 Menu De Atendimento

Pelo menu do atendente é possível realizar cadastros de conveniados, listar esses cadastros, pesquisar conveniados, deletar cadastros, agendar consultas, listar, pesquisar e deletar essas consultas, consultar a situação e deletar feedbacks e ter acesso a relatórios de faturamento e tabela.

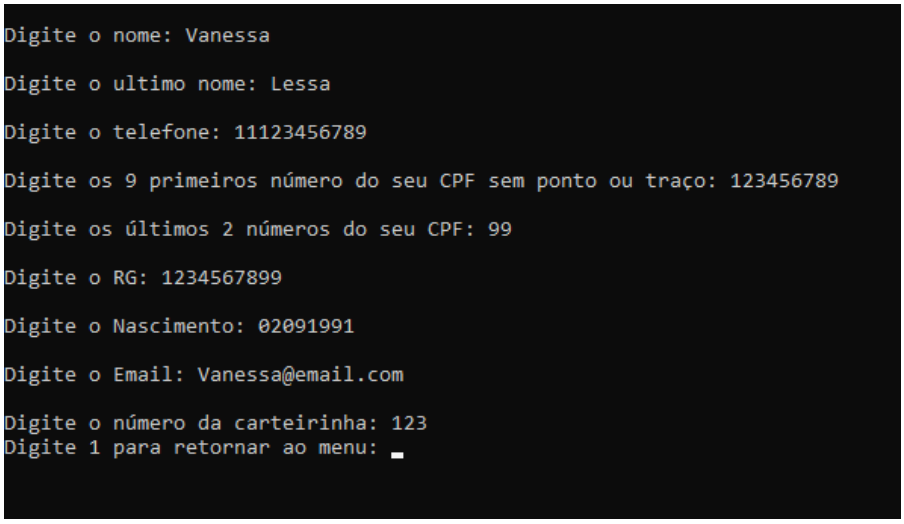
Figura **12** expõe o menu principal de atendimento



10.2.5. Cadastrando conveniado

No menu inicial, digite a opção 1 “Cadastrar Conveniado”, em seguida preencha todos os campos de cadastro e tecle “Enter” para salvar.

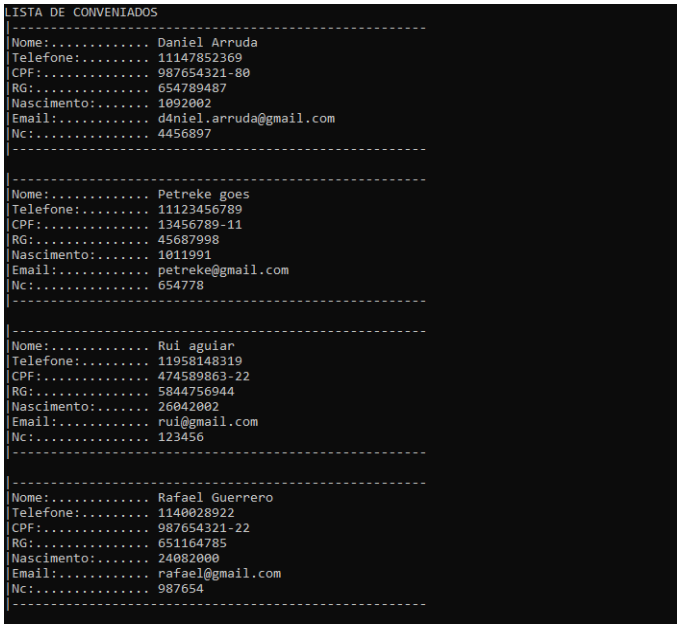
Figura **13** ostenta o cadastramento de um conveniado



10.2.6. Listando Conveniados

Para ter acesso a lista de conveniados cadastrados no sistema, digite a opção 2 “Listar Conveniados” no menu inicial, em seguida o sistema exibirá os todos os cadastros realizados no sistema.

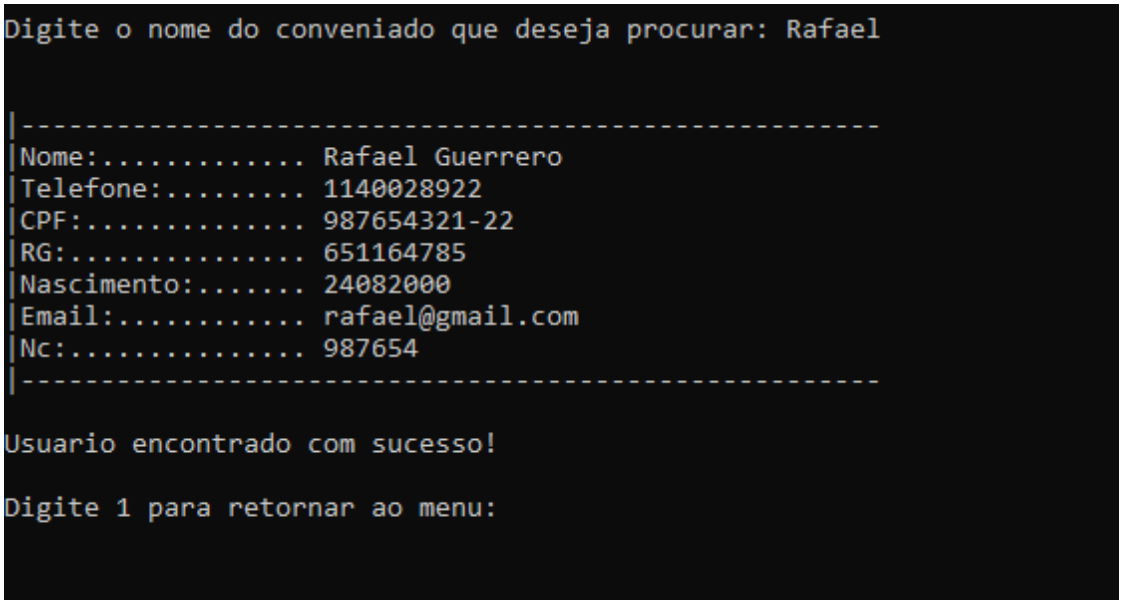
Figura **14** retrata a lista de conveniados



10.2.7. Pesquisando Conveniados

Para pesquisar um cadastro existente no sistema basta digitar a opção 3 “Pesquisar Conveniados” no menu inicial. Em seguida, digite o nome e sobrenome do conveniado e tecle “Enter”, o sistema irá exibir todos os dados cadastrados referentes ao conveniado.

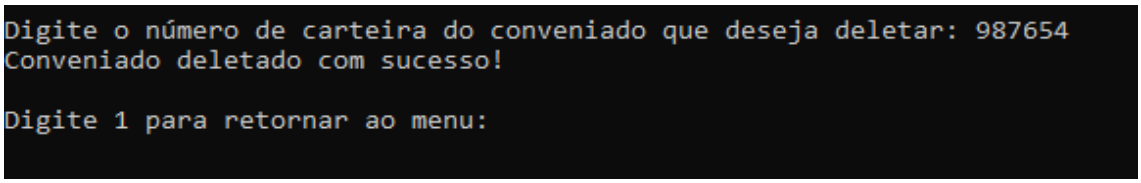
Figura **15** denota uma pesquisa de conveniado



10.2.8. Deletando Cadastro

No menu inicial, digite a opção 4 “Deletar Cadastro”. Em seguida digite o número da carteira do conveniado e tecle “Enter”.

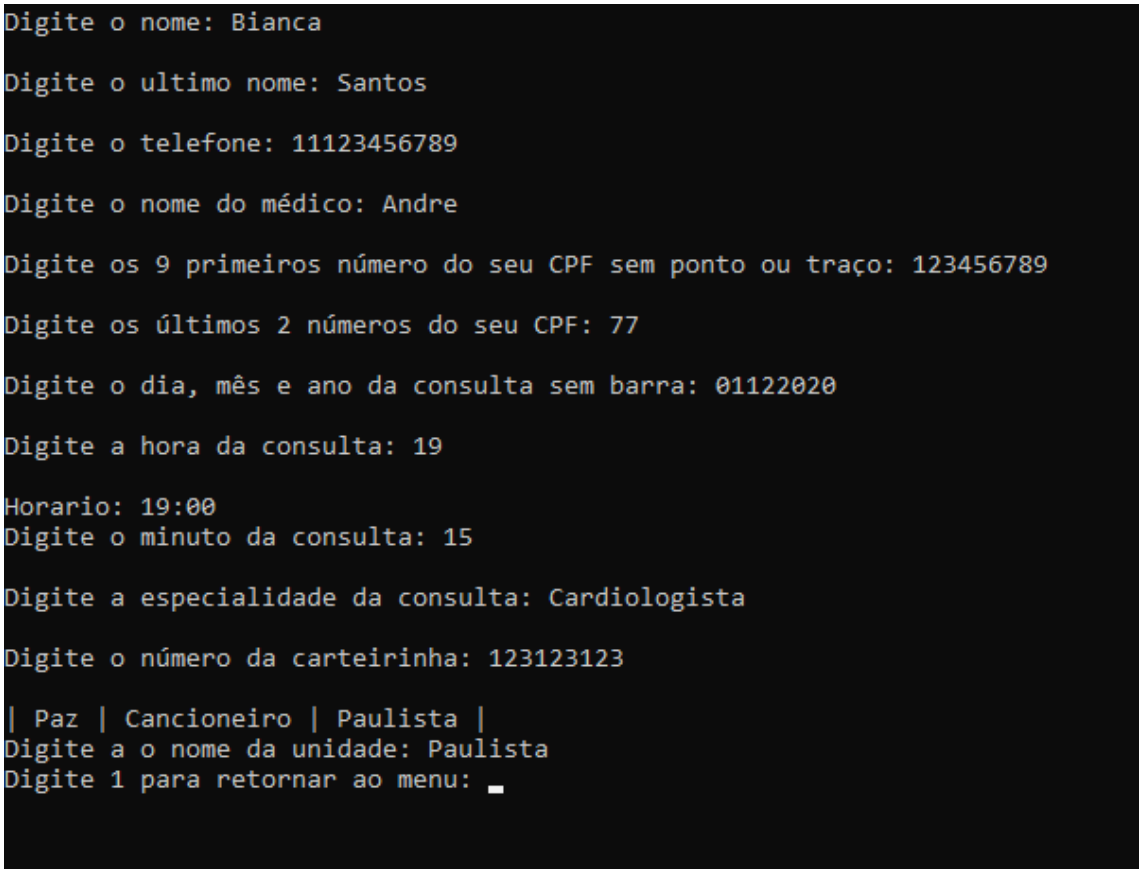
Figura **16** indica que o conveniado foi deletado com sucesso



10.2.9 Agendamento De Consultas

No menu inicial, digite a opção 5 “Agendar Consulta” e preencha todos os campos solicitados com as informações do conveniado e do agendamento, tecle “Enter” para salvar.

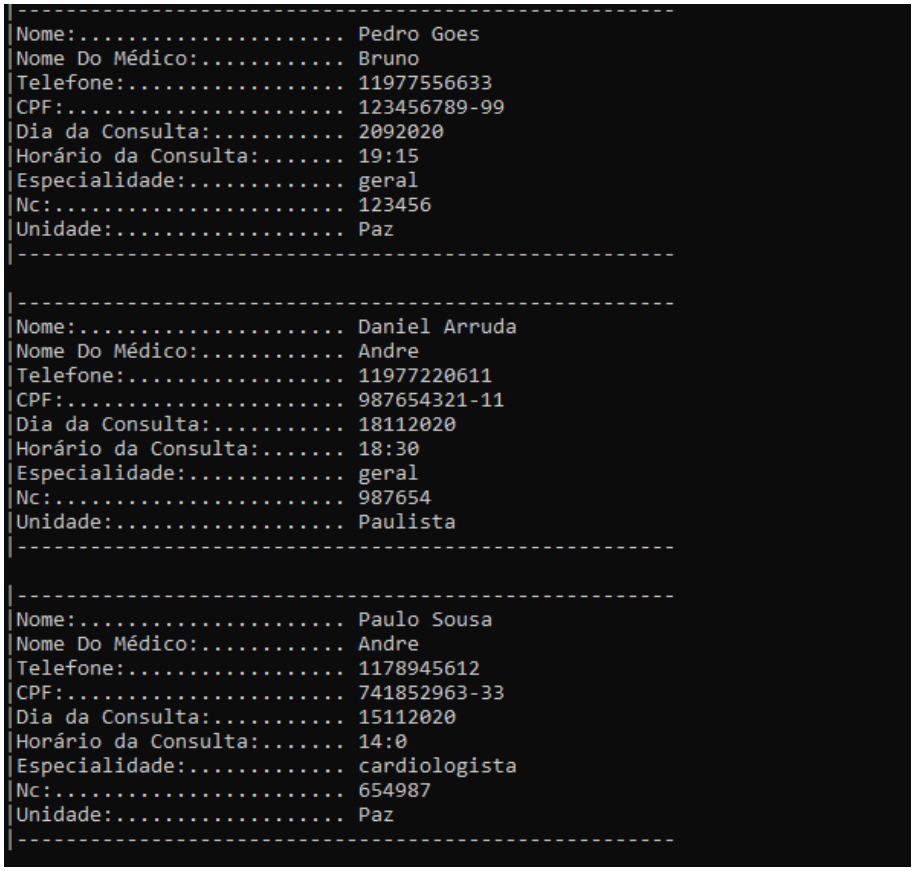
Figura **17** manifesta o agendamento de uma consulta



10.2.10 Listando Consultas

Para ter acesso as consultas já agendadas, digite a opção 6 “Listar Consultas”, e o sistema exibira todas as consultas cadastradas.

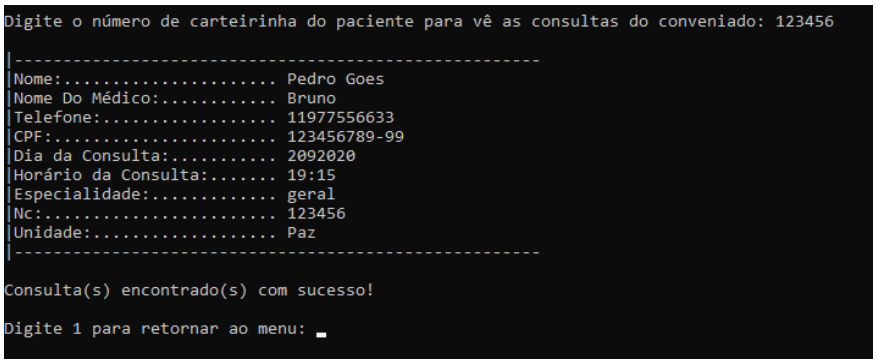
Figura **18** mostra uma lista de agendamento de consulta



10.2.11 Pesquisando Consultas

No menu inicial, digite a opção 7 “Pesquisar consultas”, em seguida digite o número da carteirinha do conveniado e tecle “Enter”.

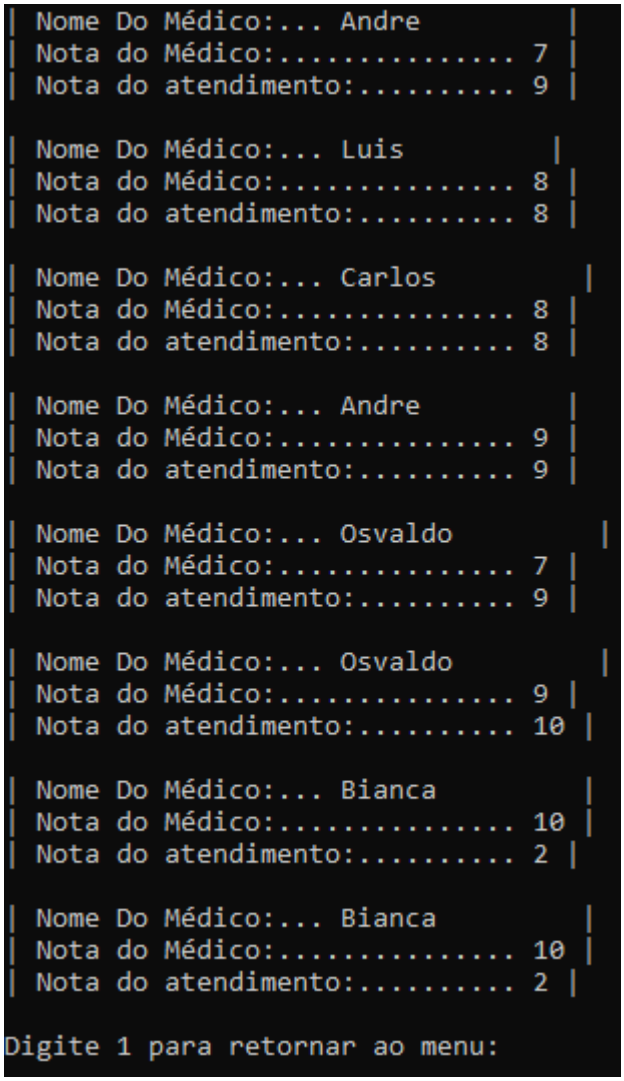
Figura **19** aponta uma pesquisa de consulta



10.2.12 Consultando Feedbacks

Para consultar a situação de feedbacks recebidos basta digitar a opção 8 “Situação de Feedbacks” no menu iniciar. Serão exibidos todos os feedbacks enviados pelos conveniados.

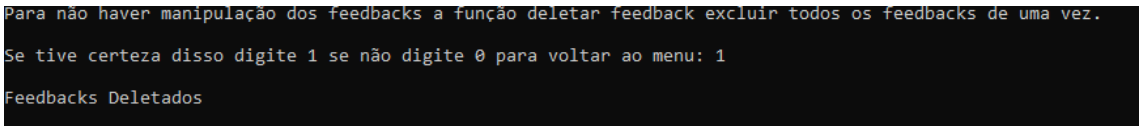
Figura **20** retrata uma consulta geral de feedback



10.2.13 Deletando Feedbacks

Para deletar feedbacks basta escolher a opção 10 “Deletar feedbacks”, é importante saber que essa operação exclui TODOS os feedbacks recebidos.

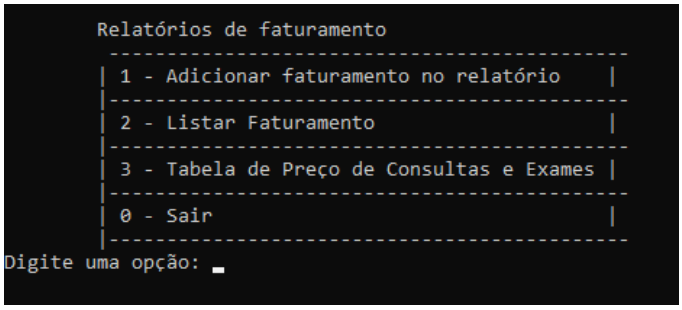
Figura **21** exibe um deletamento geral dos feedbacks



10.2.14 Relatórios De Faturamento E Tabela

Para acessar esse menu é necessário digitar a opção 11 “Relatórios de faturamento e tabela”. O seguinte menu será exibido:

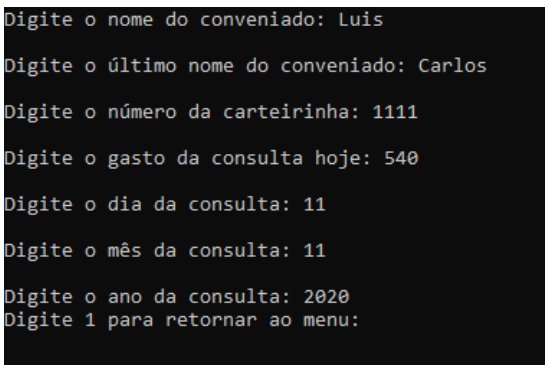
Figura **22** retrata o menu de relatórios de faturamento entro no menu de atendimento



10.2.15 Adicionando Faturamento No Relatório

Digite a opção 1 “Adicionar faturamento no relatório”, em seguida preencha todos as informações solicitadas e tecle “Enter” para adicionar e salvar.

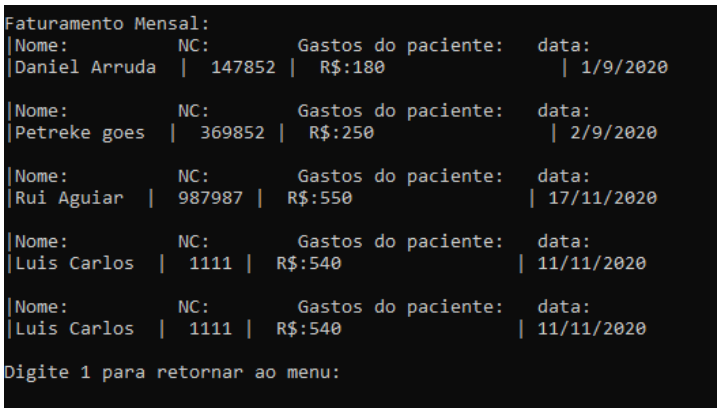
Figura **23** denota um adicionamento no relatório de faturamento



10.2.16 Listando Faturamento

Digite a opção 2 “Listar faturamento” e em seguida serão exibidos todos os registros de faturamento cadastrados.

Figura **24** mostra a lista de faturamento mensal



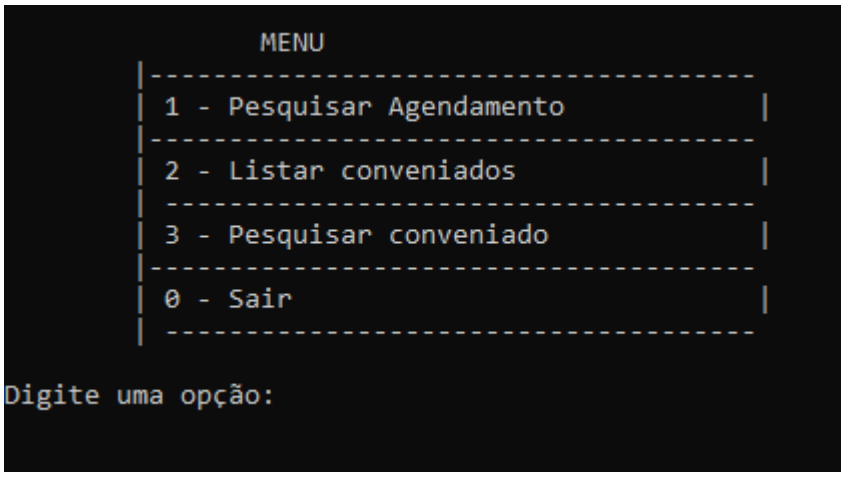
10.2.17 Tabela De Preços De Consultas E Exames

Para acessar a tabela com os respectivos valores basta digitar a opção 3 “Tabela de preços de consultas e exames” e você terá acesso a todos os valores.

Figura **25** apresenta a tabela de consulta e exames com seus preços

No menu do médico é possível pesquisar agendamentos e listar e pesquisar conveniados.

Figura **26** aponta o menu de médico



10.2.18 Pesquisando Agendamentos

Para ter acesso as consultas já agendadas, digite a opção 1 “Pesquisar agendamentos”, e o sistema exibira todas as consultas agendadas.

Figura **27** expressa uma pesquisa de agendamento no menu de médico

```
|-----------------------------------------------------
|Nome:..................... Pedro Goes
|Nome Do Médico:........... Bruno
|Telefone:................. 11977556633
|CPF:...................... 123456789-99
|Dia da Consulta:.......... 2092020
|Horário da Consulta:...... 19:15
|Especialidade:............ geral
|Nc:....................... 123456
|Unidade:.................. Paz
|-----------------------------------------------------

|-----------------------------------------------------
|Nome:..................... Daniel Arruda
|Nome Do Médico:........... Andre
|Telefone:................. 11977220611
|CPF:...................... 987654321-11
|Dia da Consulta:.......... 18112020
|Horário da Consulta:...... 18:30
|Especialidade:............ geral
|Nc:....................... 987654
|Unidade:.................. Paulista
|-----------------------------------------------------

|-----------------------------------------------------
|Nome:..................... Paulo Sousa
|Nome Do Médico:........... Andre
|Telefone:................. 1178945612
|CPF:...................... 741852963-33
|Dia da Consulta:.......... 15112020
|Horário da Consulta:...... 14:0
|Especialidade:............ cardiologista
|Nc:....................... 654987
|Unidade:.................. Paz
|-----------------------------------------------------
```

10.2.19 Pesquisando Conveniados

Para pesquisar um cadastro existente no sistema basta digitar a opção 3 “Pesquisar Conveniado” no menu inicial. Em seguida, digite o nome e sobrenome do conveniado e tecle “Enter”, o sistema irá exibir todos os dados cadastrados referentes ao conveniado.

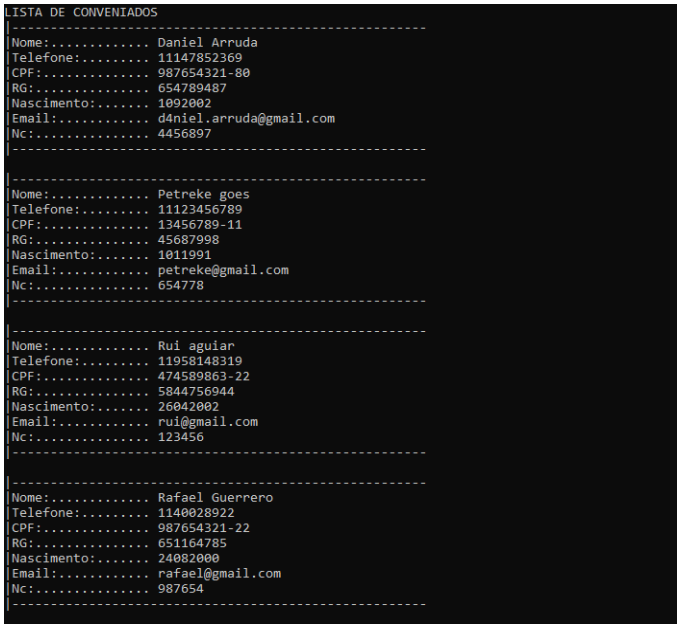
Figura **28** expõe uma pesquisa de consulta no menu médico

```
Digite o nome do conveniado que deseja procurar: Rafael


|-----------------------------------------------------
|Nome:............. Rafael Guerrero
|Telefone:......... 1140028922
|CPF:.............. 987654321-22
|RG:............... 651164785
|Nascimento:....... 24082000
|Email:............ rafael@gmail.com
|Nc:............... 987654
|-----------------------------------------------------

Usuario encontrado com sucesso!

Digite 1 para retornar ao menu:
```

10.2.20 Listando Conveniados

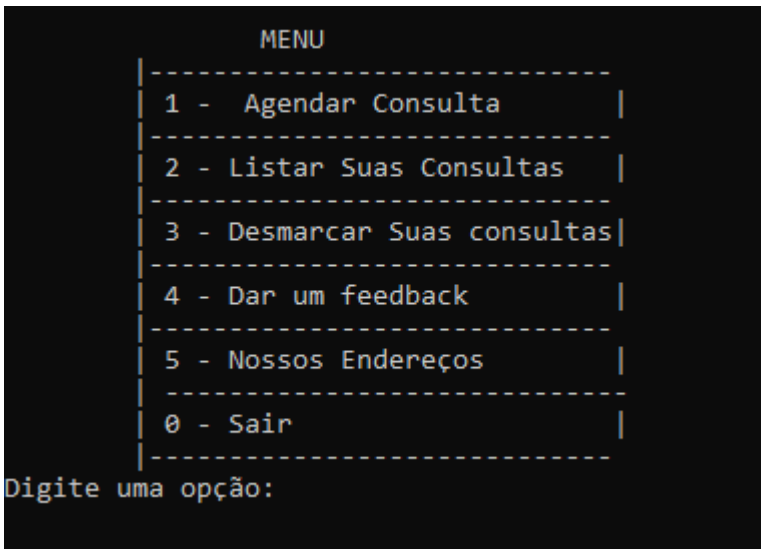
Para ter acesso a lista de conveniados cadastrados no sistema, digite a opção 2 “Listar Conveniados” no menu inicial, em seguida o sistema exibirá os todos os cadastros realizados no sistema.

Figura **29** lista os conveniados no menu de médico



No menu do conveniado é possível agendar, listar e desmarcar consultas, enviar feedbacks e consultar os endereços de todas as nossas unidades.

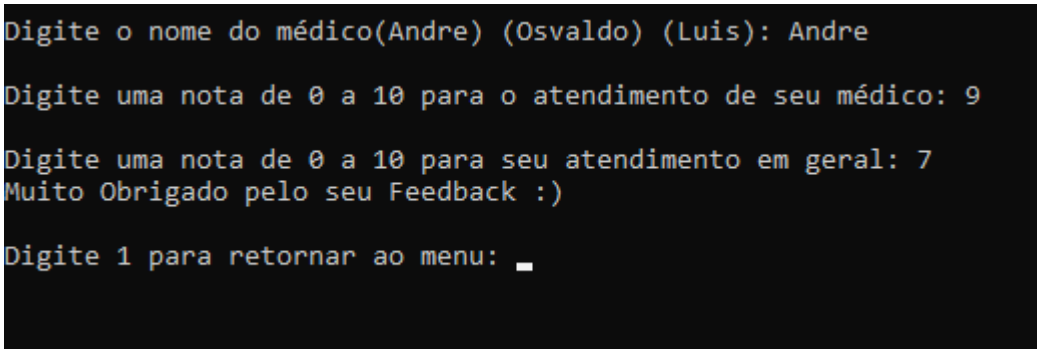
Figura **30** exibe o menu do conveniado



10.2.21 Enviando Feedback

Para dar feedback a algum funcionário basta digitar a opção 4 “Dar um feedback”, em seguida basta digitar o nome do funcionário, digitar uma nota para o atendimento do médico e o atendimento em geral e apertar “Enter” para enviar.

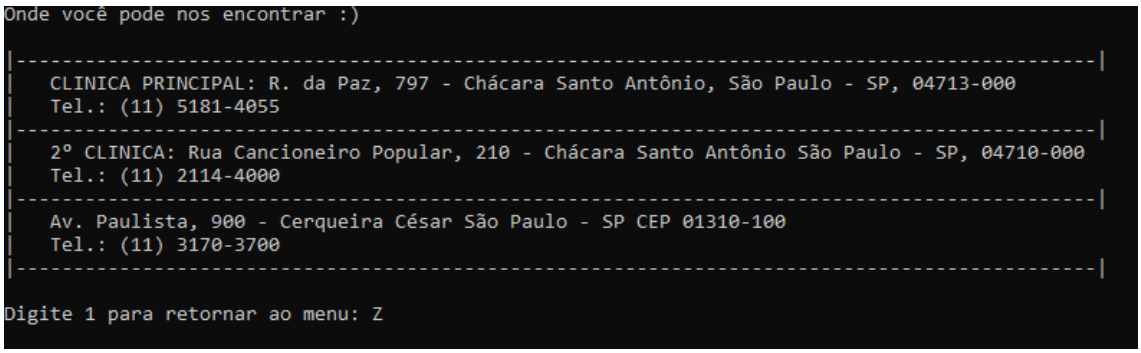
Figura **31** expõe o enviamento de um feedback feito pelo conveniado



10.2.22 Consultando Nossos Endereços

Para consultar os endereços das nossas clínicas, basta digitar a opção 5 “Nossos endereços” e o sistema apresentará todos os nossos locais de atendimento.

Figura **32** mostra os endereços das clínicas



10.3. APÊNDICE C - FLUXOGRAMAS

Figura **33** mostra o fluxograma da tela inicial (de login) e suas possibilidade

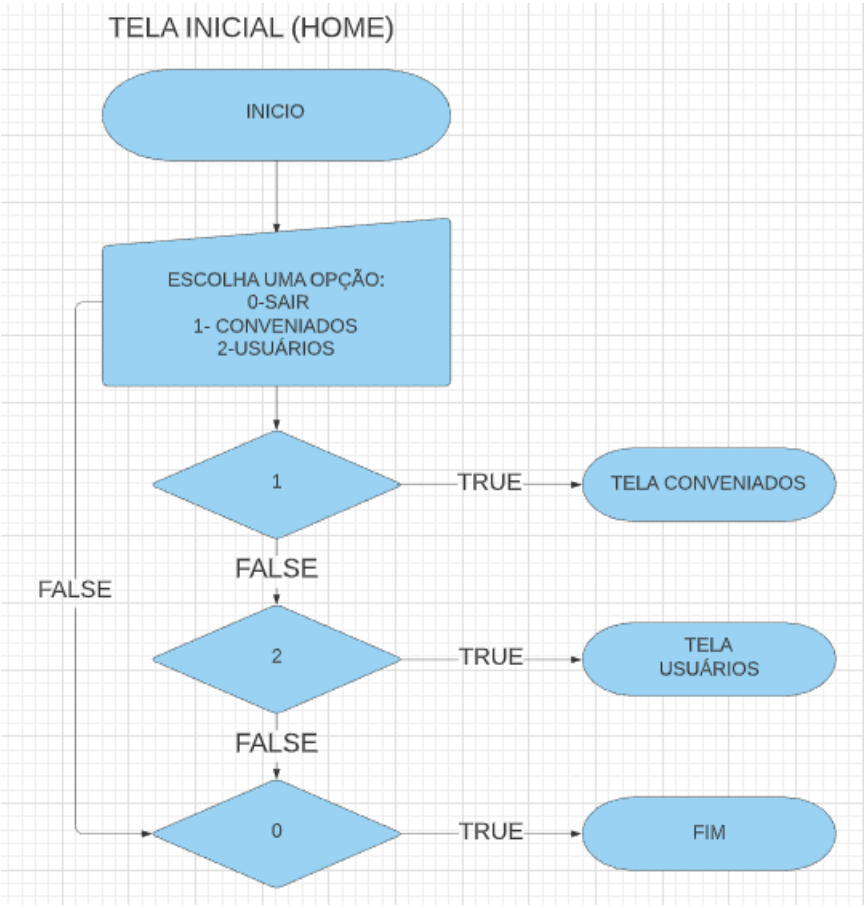


Figura **34** as possibilidades da tela de conveniado

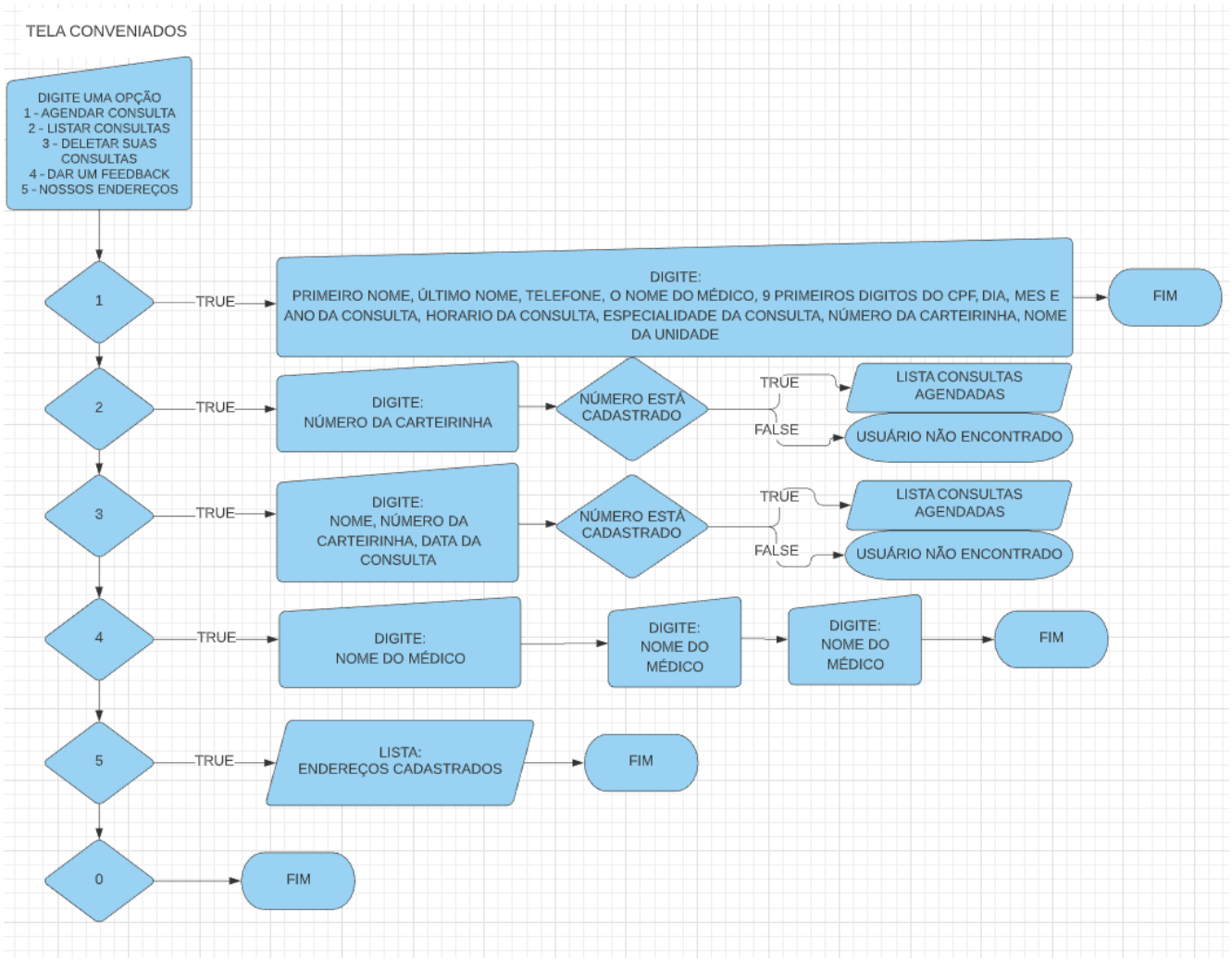
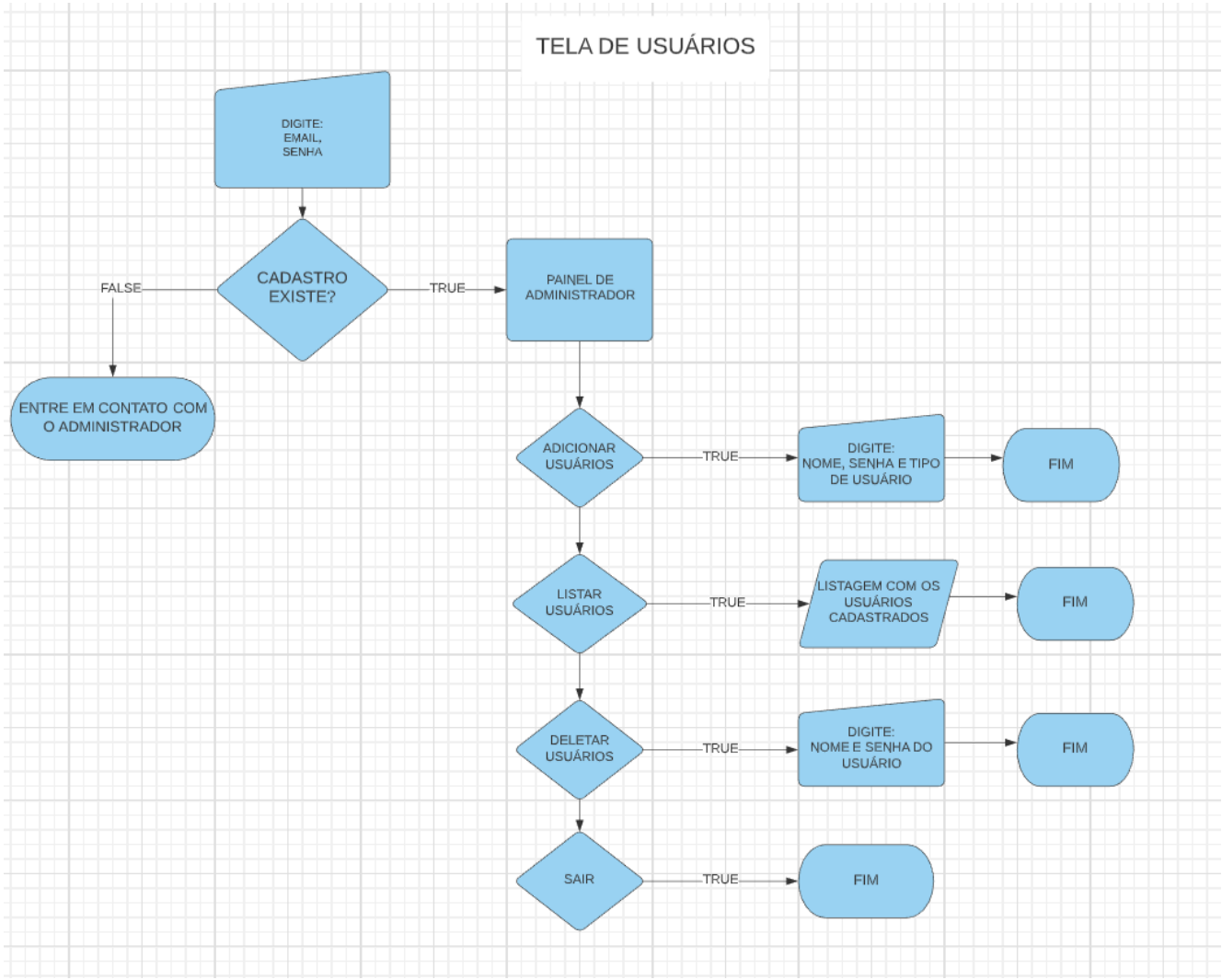


Figura **35** apresenta o fluxograma de login e cadastramento de usuários pela conta de administrador



10.4. APÊNDICE D - CÓDIGO DO SISTEMA EM LINGUAGEM C

```
#include <stdio.h>
#include <string.h>
#include <locale.h>
#include <stdbool.h>
#include <TIME.H>

//variaveis globais
int volta_menu; //variável global para retornar ao menu


//função para abrir arquivo
FILE* AbreArquivo(char modo, char caminho[30]);
FILE* AbreArquivo(char modo, char caminho[30])
{
   FILE *arquivo;
   switch(modo)
   {
   case 'g':
      arquivo = fopen(caminho,"wt"); //abre o arquivo em modo de gravação
      break;
   case 'l':
      arquivo = fopen(caminho,"rt"); //abre o arquivo em modo de leitura
      break;
   case 'a':
      arquivo = fopen(caminho,"a"); //abre o arquivo em um modo append
      break;
   }
   if(arquivo==NULL)  //Se houver algum erro, o ponteiro apontará para NULL
   {
      printf("Nao foi possível abrir o arquivo");
      exit(0);
   }
   return arquivo;
}

//void para fechar arquivo
void FecharArquivo(FILE *arquivo);
void FecharArquivo(FILE *arquivo)
{
   fclose(arquivo); //apenas usa a função para fechar o arquivo com o ponteiro arquivo
}

// Função para Cadastrar Feedbacks
```

```
void CadastraFeedback(char nome_medico[30], int nota_medico, int
nota_atendimento);
void CadastraFeedback(char nome_medico[30], int nota_medico, int
nota_atendimento)
{

   setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
   FILE *arquivo;

   arquivo = AbreArquivo('a', "Listafeedback.txt");//Abre o arquivo Lista de Feedbacks
em modo append
   fprintf(arquivo, "%s %d %d \n", nome_medico, nota_medico,
nota_atendimento);//Cadastra as variáveis no arquivo texto
   FecharArquivo(arquivo);//fecha arquivo
}

//função para Listar Feedbacks
void ListarFeedback();
void ListarFeedback()
{

   setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
   //Variáveis e ponteiros
   FILE *arquivo;
   char nome_medico[30];
   int nota_medico;
   int nota_atendimento;

   arquivo = AbreArquivo('l',"ListaFeedback.txt");//Abre o arquivo Lista de Feedbacks
em modo leitura
   while(!feof(arquivo))//Usa o loop para ler o arquivo todo com o ponteiro *arquivo
   {
      fscanf(arquivo,"%s %d %d",&nome_medico, &nota_medico,
&nota_atendimento);//Escaneia todos as tudo que tem e adiciona nas variáveis
      printf("| Nome Do Médico:... %s        |\n| Nota do Médico:..... %d |\n| Nota do
atendimento:..... %d |\n\n",nome_medico, nota_medico, nota_atendimento);//Printa
na tela a Lista de Feedback
   }
   FecharArquivo(arquivo);//fecha arquivo

}

//função para Deletar Feedbacks
void DeletarFeedback();
void DeletarFeedback()
{
```

```
    setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
    FILE *arquivo;//Ponteiro

    remove("ListaFeedback.txt");//Deleta o arquivo Lista de Feedback
    remove("ListaFeedbackTemp.txt");//Deleta o arquivo Lista de Feedback Temp

    arquivo = AbreArquivo('g', "Listafeedback.txt");//Cria o arquivo Lista de Feedback
novamente
    FecharArquivo(arquivo);//Fecha arquivo

    printf("\nFeedbacks Deletados\n\n");//Avisa que os feedbacks foram deletados
    system("pause");
}

// função para Agendar consultas
void Agendar(char nome[30], char ult_nome[30], char nome_medico[30], char
telefone[20], char cpf_sdigito[15], char cpf_digito[15], int dia_consulta, int
hora_consulta, int minuto_consulta, char especialidade[30], char num_carteira[30],
char unidade[20]);
void Agendar(char nome[30], char ult_nome[30], char nome_medico[30], char
telefone[20], char cpf_sdigito[15], char cpf_digito[15], int dia_consulta, int
hora_consulta, int minuto_consulta, char especialidade[30], char num_carteira[30],
char unidade[20])
{
    //ponteiro
    FILE *arquivo;

    arquivo = AbreArquivo('a', "ListaAgendamento.txt");//Abre o arquivo Lista
Agendamento em modo append
    fprintf(arquivo, "%s %s %s %s %s %s %d %d %d %s %s %s\n", nome, ult_nome,
nome_medico, telefone, cpf_sdigito, cpf_digito, dia_consulta, hora_consulta,
minuto_consulta, especialidade, num_carteira, unidade);//Escreve o foi pedido e estar
nas variáveis para o arquivo texto
    FecharArquivo(arquivo);//Fecha arquivo
}

// função para Listar consultas
void ListarAgenda();
void ListarAgenda()
{

    setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
    //Ponteiros e Variáveis
    FILE *arquivo;
    char nome[50];
```

```
    char ult_nome[50];
    char nome_medico[30];
    char numAnterior[50];
    char telefone[20];
    char cpf_sdigito[15];
    char cpf_digito[15];
    int dia_consulta;
    int hora_consulta;
    int minuto_consulta;
    char especialidade[30];
    char num_carteira[30];
    char unidade[20];

    arquivo = AbreArquivo('l',"ListaAgendamento.txt");//Abre arquivo Lista
Agendamento em modo append
    while(!feof(arquivo))//Usa o loop para ler o arquivo todo com o ponteiro *arquivo
    {
        fscanf(arquivo,"%s %s %s %s %s %s %d %d %d %s %s %s", &nome, &ult_nome,
&nome_medico, &telefone, &cpf_sdigito, &cpf_digito, &dia_consulta, &hora_consulta,
&minuto_consulta, &especialidade, &num_carteira, &unidade);//Escaneia todos as
tudo que tem e adiciona nas variáveis
        if(strcmp(numAnterior,num_carteira) != 0)//Compara os números de carteira para
não dar um bug que aparecer a mesma consulta 2 vezes
        {
            strcpy(numAnterior,num_carteira);//Copia a variável numAnterior no e cola em
numCarteira para não ocorrer o bug que ira repetir 2 vezes a consulta
            printf("|-----------------------------------------------------\n|Nome:..... %s
%s\n|Nome Do Médico:..... %s\n|Telefone:..... %s\n|CPF:..... %s-
%s\n|Dia da Consulta:..... %d\n|Horário da Consulta:.....
%d:%d\n|Especialidade:..... %s\n|Nc:..... %s\n|Unidade:.....
%s\n|-----------------------------------------------------\n\n", nome, ult_nome, nome_medico,
telefone, cpf_sdigito, cpf_digito, dia_consulta, hora_consulta, minuto_consulta,
especialidade, num_carteira, unidade);//printa todas as informações de agendamento
        }
    }
    FecharArquivo(arquivo);//Fecha arquivo
}

//função para pesquisar consulta
void PesquisaAgendamento (char pesquisa_num[30]);
void PesquisaAgendamento (char pesquisa_num[30])
{
    setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
    //ponteiros e variáveis
    FILE *arquivo;
    FILE *arquivoTemp;
```

```
   char nome[50];
   char ult_nome[50];
   char telefone[20];
   char nome_medico[30];
   char cpf_sdigito[15];
   char cpf_digito[15];
   int dia_consulta;
   int hora_consulta;
   int minuto_consulta;
   char especialidade[30];
   char num_carteira[30];
   char numAnterior[50];
   char nomeAnterior[50];
   char unidade[20];
   bool achou = false;//Coloca essa variável em FALSE para ter uma confirmação de deletado mais a frente

   arquivo = AbreArquivo('l', "ListaAgendamento.txt");//Abre o arquivo Lista Agendamento em modo de leitura
   arquivoTemp = AbreArquivo('g', "PesquisaTemp.txt");//Abre o arquivo Lista Agendamento em modo de gravação
   while(!feof(arquivo))//Usa o loop para ler o arquivo todo com o ponteiro *arquivo
   {
      fscanf(arquivo,"%s %s %s %s %s %s %d %d %d %s %s %s", &nome, &ult_nome, &nome_medico, &telefone, &cpf_sdigito, &cpf_digito, &dia_consulta, &hora_consulta, &minuto_consulta, &especialidade, &num_carteira, &unidade);//Escaneia tudo que estar em Lista Agendamento e joga nas variáveis
      // Zero = são iguais
      // Diferente de Zero = não são iguais
      if(strcmp(nomeAnterior,nome) != 0)//Compara os nomes para não dar um bug que aparecer a mesma consulta 2 vezes
      {
         strcpy(nomeAnterior,nome);//Copia a variável nomeAnterior no e cola em nome para não ocorrer o bug que ira repetir 2 vezes a consulta
         if(strcmp(pesquisa_num,num_carteira) == 0)//Compara o número de carteira, se for igual printa no arquivo Temp a linha toda
         {
            fprintf(arquivoTemp, "%s %s %s %s %s %s %d %d %d %s %s %s\n", nome, ult_nome, nome_medico, telefone, cpf_sdigito, cpf_digito, dia_consulta, hora_consulta, minuto_consulta, especialidade, num_carteira, unidade);//printa a linha que seja igual os parâmetros
         }
         else//Se não ele coloca a variável "achou" em true
         {
            achou = true;
         }
```

```
      }
   }
   FecharArquivo(arquivo);//Fecha arquivo
   FecharArquivo(arquivoTemp);//Fecha arquivo Temp


   arquivo = AbreArquivo('l',"PesquisaTemp.txt");//Abre arquivo Pesquisa Temp em modo leitura

   strcpy(nomeAnterior,"");//Copia nomeAnterior para "" para a variável ficar vazia
   while(!feof(arquivo))//Usa o loop para ler o arquivo todo com o ponteiro *arquivo
   {
      fscanf(arquivo,"%s %s %s %s %s %s %d %d %d %s %s %s", &nome, &ult_nome, &nome_medico, &telefone, &cpf_sdigito, &cpf_digito, &dia_consulta, &hora_consulta, &minuto_consulta, &especialidade, &num_carteira, unidade);//Escaneia tudo que estar em Lista Agendamento e joga nas variáveis

      if(strcmp(nomeAnterior,nome) != 0 && strcmp(pesquisa_num,num_carteira) == 0)//Se nomeAnterior for diferente de nome e pesquisa_num por igual a num_carteira ele irá printa as informações de pesquisa encontrados
      {
         strcpy(nomeAnterior,nome);
         printf("\n|-----------------------------------------------------\n|Nome:..... %s %s\n|Nome Do Médico:..... %s\n|Telefone:..... %s\n|CPF:..... %s-%s\n|Dia da Consulta:..... %d\n|Horário da Consulta:..... %d:%d\n|Especialidade:..... %s\n|Nc:..... %s\n|Unidade:..... %s\n|-----------------------------------------------------\n\n", nome, ult_nome, nome_medico, telefone, cpf_sdigito, cpf_digito, dia_consulta, hora_consulta, minuto_consulta, especialidade, num_carteira, unidade);
         printf("Consulta(s) encontrado(s) com sucesso!\n\n");
      }
      if(strcmp(pesquisa_num,num_carteira) !=0)//se pesquisa_num for diferente de num_carteira ira printa Usuario não encontrado!
      {
         printf("Usuario não encontrado!\n\n");
      }
   }
}

// função para Deletar consultas //pronta
void DeletarAgendamento(char nomeDeletarAgenda[50], char num_carteiraDeletar[30], int diaDeletar, int horarioDeletar, int minutoDeletar);
void DeletarAgendamento(char nomeDeletarAgenda[50], char num_carteiraDeletar[30], int diaDeletar, int horarioDeletar, int minutoDeletar)
{
   setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
```

```
    //Ponteiros e Variáveis
    FILE *arquivo;
    FILE *arquivoTemp;
    char nome[50];
    char ult_nome[50];
    char telefone[20];
    char nome_medico[30];
    char cpf_sdigito[15];
    char cpf_digito[15];
    int dia_consulta;
    int hora_consulta;
    int minuto_consulta;
    char especialidade[30];
    char num_carteira[30];
    char numAnterior[50];
    char unidade[20];
    bool apagou = false;//Coloca essa variável em FALSE para ter uma confirmação de deletado mais a frente

    arquivo = AbreArquivo('l', "ListaAgendamento.txt");//Abre o arquivo Lista Agendamento em modo leitura
    arquivoTemp = AbreArquivo('g', "ListaAgendamentoTemp.txt");//Abre o arquivo Lista Agendamento em modo gravação

    while(!feof(arquivo))//Usa o loop para ler o arquivo todo com o ponteiro *arquivo
    {
        fscanf(arquivo,"%s %s %s %s %s %s %d %d %d %s %s %s", &nome, &ult_nome, &nome_medico, &telefone, &cpf_sdigito, &cpf_digito, &dia_consulta, &hora_consulta, &minuto_consulta, &especialidade, &num_carteira, &unidade);//Escaneia tudo que estar em Lista Agendamento e joga nas variáveis
        // Zero = são iguais
        // Diferente de Zero = não são iguais
        if(strcmp(numAnterior,num_carteira) != 0)//Compara os nomes para não dar um bug que deleta 2 consultas
        {
            strcpy(numAnterior,num_carteira);//Copia a variável numAnterior no e cola em num_carteira para não ocorrer o bug descrito acima
            if(strcmp(nomeDeletarAgenda,nome) != 0 && strcmp(num_carteiraDeletar,num_carteira) != 0 && diaDeletar != dia_consulta && horarioDeletar != hora_consulta && minutoDeletar != minuto_consulta)//compara os parâmetros do If para confirmar se é a consulta certa antes de apagar
            {
                fprintf(arquivoTemp, "%s %s %s %s %s %s %d %d %d %s %s %s\n", nome, ult_nome, nome_medico, telefone, cpf_sdigito, cpf_digito, dia_consulta, hora_consulta, minuto_consulta, especialidade, num_carteira, unidade);//printa todas as consulta menos a que o usuário deseja apagar em Pesquisa Agendamento Temp
```

```
        }
        else
        {
           apagou = true;//se der algum erro apagou vira True
        }
      }
   }
   FecharArquivo(arquivo);//fecha arquivo
   FecharArquivo(arquivoTemp);//fecha arquivo Temp

   arquivo = AbreArquivo('g', "ListaAgendamento.txt");//Abre arquivo Lista
Agendamento em modo gravação
   arquivoTemp = AbreArquivo('l', "ListaAgendamentoTemp.txt");//Abre arquivo Lista
Agendamento em modo de leitura

   strcpy(numAnterior,"");//Copia numAnterior para "NULL" para a variável ficar vazia

   while(!feof(arquivoTemp))//Usa o loop para ler o arquivo todo com o ponteiro
*arquivoTemp
   {
      fscanf(arquivoTemp,"%s %s %s %s %s %s %d %d %d %s %s %s", &nome,
&ult_nome, &nome_medico, &telefone, &cpf_sdigito, &cpf_digito, &dia_consulta,
&hora_consulta, &minuto_consulta, &especialidade, &num_carteira,
&unidade);//Escaneia tudo que estar em Lista Agendamento e joga nas variáveis
      if(strcmp(numAnterior,num_carteira))//Compara numAnterior com num_carteira
      {
         strcpy(numAnterior,num_carteira);//Copia a variável numAnterior no e cola em
num_carteira
         fprintf(arquivo, "%s %s %s %s %s %s %d %d %d %s %s %s\n", nome,
ult_nome, nome_medico, telefone, cpf_sdigito, cpf_digito, dia_consulta,
hora_consulta, minuto_consulta, especialidade, num_carteira, unidade);//Printa todas
as consultas menos a que foi deletada
      }
   }

   FecharArquivo(arquivo);//Fecha Arquivo
   FecharArquivo(arquivoTemp);//Fecha arquivoTemp

   if(apagou)//Se apagou virou true ele irá printar que o a consulta foi apagada com
sucesso
   {
      printf("Agendamento de Consulta apagada com sucesso!\n\n");
   }
   else//Se apagou for falso ele irá printar que o conveniado não foi encontrado
   {
      printf("Conveniado não encontrado!\n\n");
```

```
    }
}

//função cadastra conveniado
void Cadastra(char nome[30], char ult_nome[30], char telefone[20], char
cpf_sdigito[15], char cpf_digito[15], char rg[15], int nascimento, char email[40], char
num_carteira[30]);
void Cadastra(char nome[30], char ult_nome[30], char telefone[20], char
cpf_sdigito[15], char cpf_digito[15], char rg[15], int nascimento, char email[40], char
num_carteira[30])
{
    //Ponteiros
    FILE *arquivo;

    arquivo = AbreArquivo('a', "ListaCadastro.txt");//Abre o arquivo Lista Cadastro em
modo de append
    fprintf(arquivo, "%s %s %s %s %s %s %d %s %s\n", nome, ult_nome, telefone,
cpf_sdigito, cpf_digito, rg, nascimento, email, num_carteira);//Escreve o foi pedido e
estar nas variáveis para o arquivo texto
    FecharArquivo(arquivo);//Fecha arquivo
}

//função para listar conveniados //pronta
void Listar();
void Listar()
{

    setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
    //Ponteiros e variáveis
    FILE *arquivo;
    char nome[50];
    char ult_nome[50];
    char nomeAnterior[50];
    char telefone[20];
    char cpf_sdigito[15];
    char cpf_digito[15];
    char rg[15];
    int nascimento;
    char email[40];
    char num_carteira[30];

    printf("LISTA DE CONVENIADOS");
    arquivo = AbreArquivo('l',"ListaCadastro.txt");//Abre o arquivo Lista Cadastro em
modo de leitura
    while(!feof(arquivo))//Usa o loop para ler o arquivo todo com o ponteiro *arquivo
```

```
    {
        fscanf(arquivo,"%s %s %s %s %s %s %d %s %s", &nome, &ult_nome, &telefone, &cpf_sdigito, &cpf_digito, &rg, &nascimento, &email, &num_carteira);//Escaneia tudo que tem e adiciona nas variáveis
        if(strcmp(nomeAnterior,nome) != 0)//Compara o nomeAnterior com o nome para não dar um bug que aparecer o mesmo conveniado 2 vezes
        {
            strcpy(nomeAnterior,nome);//Copia a variável nomeAnteiror e cola em nome para não ocorrer o bug que ira repetir 2 vezes a consulta
            printf("\n|-----------------------------------------------------\n|Nome:..... %s %s\n|Telefone:..... %s\n|CPF:..... %s-%s\n|RG:..... %s\n|Nascimento:..... %d\n|Email:..... %s\n|Nc:..... %s\n|-----------------------------------------------------\n", nome, ult_nome, telefone, cpf_sdigito, cpf_digito, rg, nascimento, email, num_carteira);
        }
    }
    FecharArquivo(arquivo);//Fecha arquivo
}

//função para pesquisar conveniado
void PesquisaConveniado (char pesquisa_nome[50]);
void PesquisaConveniado (char pesquisa_nome[50])
{

    setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
    //Ponteiros e variáveis
    FILE *arquivo;
    FILE *arquivoTemp;
    char nome[50];
    char ult_nome[50];
    char telefone[20];
    char cpf_sdigito[15];
    char cpf_digito[15];
    char rg[15];
    int nascimento;
    char email[40];
    char num_carteira[30];
    char numAnterior[50];
    bool achou = false;

    arquivo = AbreArquivo('l', "ListaCadastro.txt");//Abre o arquivo Lista Cadastro em modo de leitura
    arquivoTemp = AbreArquivo('g', "PesquisaTemp.txt");//Abre o arquivo Lista Cadastro em modo de gravação
    while(!feof(arquivo))//Usa o loop para ler o arquivo todo com o ponteiro *arquivo
    {
```

```
      fscanf(arquivo,"%s %s %s %s %s %s %d %s %s", &nome, &ult_nome, &telefone,
&cpf_sdigito, &cpf_digito, &rg, &nascimento, &email, &num_carteira);//Escaneia tudo
que tem e adiciona nas variáveis
      // Zero = são iguais
      // Diferente de Zero = não são iguais
      if(strcmp(numAnterior,num_carteira) != 0)//Compara numAnterior com
num_carteira para não ocorre o erro que repetirar 2 vezes o ultimo conveniado
      {
         strcpy(numAnterior,num_carteira);//Se o IF for diferente ele ira copiar o
numAnterior e cola no num_carteira
         if(strcmp(pesquisa_nome,nome) == 0)//Compara o pesquisa_nome com o
nome, se for igual ele irar printar em pesquisa Temp
         {
            fprintf(arquivoTemp, "%s %s %s %s %s %s %d %s %s\n", nome, ult_nome,
telefone, cpf_sdigito, cpf_digito, rg, nascimento, email, num_carteira);
         }
         else
         {
            achou = true;
         }
      }
   }
   FecharArquivo(arquivo);//Fecha arquivo
   FecharArquivo(arquivoTemp);//Fecha arquivoTemp


   arquivo = AbreArquivo('l',"PesquisaTemp.txt");//Abre o arquivo Pesquisa Temp em
modo leitura

   strcpy(numAnterior,"");//Copia nomeAnterior para "NULL" para a variável ficar vazia
   while(!feof(arquivo))//Usa o loop para ler o arquivo todo com o ponteiro *arquivo
   {
      fscanf(arquivo,"%s %s %s %s %s %s %d %s %s", &nome, &ult_nome, &telefone,
&cpf_sdigito, &cpf_digito, &rg, &nascimento, &email, &num_carteira);//Escaneia tudo
que tem e adiciona nas variáveis
      if(strcmp(numAnterior,num_carteira) != 0 && strcmp(pesquisa_nome,nome) == 0)
      {
         strcpy(numAnterior,num_carteira);//Copia a variável nomeAnterior no e cola em
nome para não ocorrer o bug que copia 2 vezes
         printf("\n\n|-----------------------------------------------------\n|Nome:..... %s
%s\n|Telefone:..... %s\n|CPF:..... %s-%s\n|RG:..... %s\n|Nascimento:.....
%d\n|Email:..... %s\n|Nc:..... %s\n|-----------------------------------------------------\n\n",
nome, ult_nome, telefone, cpf_sdigito, cpf_digito, rg, nascimento, email,
num_carteira);
         printf("Usuario encontrado com sucesso!\n\n");
      }
```

```
        if(strcmp(pesquisa_nome,nome) !=0)//compara o pesquisa_nome e nome, se for diferente ele printa usuário não encontrado
        {
            printf("Usuário não encontrado!\n\n");
        }
    }
}

//função para deletar conveniado
void Deletar(char num_Deletar[50]);
void Deletar(char num_Deletar[50])
{

    setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
    //Ponteiros e Variáveis
    FILE *arquivo;
    FILE *arquivoTemp;
    char nome[50];
    char ult_nome[50];
    char telefone[20];
    char cpf_sdigito[15];
    char cpf_digito[15];
    char rg[15];
    int nascimento;
    char email[40];
    char num_carteira[30];
    char nomeAnterior[50];
    bool apagou = false;//Coloca essa variável em FALSE para ter uma confirmação de deletado mais a frente

    arquivo = AbreArquivo('l', "ListaCadastro.txt");//Abre o arquivo Lista Cadastro em modo leitura
    arquivoTemp = AbreArquivo('g', "ListaCadastroTemp.txt");//Abre o arquivo Lista Cadastro Temp em modo gravação

    while(!feof(arquivo))//Usa o loop para ler o arquivo todo com o ponteiro *arquivo
    {
        fscanf(arquivo,"%s %s %s %s %s %s %d %s %s", &nome, &ult_nome, &telefone, &cpf_sdigito, &cpf_digito, &rg, &nascimento, &email, &num_carteira);//Escaneia tudo que estar em Lista Cadastro e joga nas variáveis
        // Zero = são iguais
        // Diferente de Zero = não são iguais
        if(strcmp(nomeAnterior,nome) != 0)//Compara os nomes para não dar um bug que deleta 2 consultas
        {
```

```
        strcpy(nomeAnterior,nome);//Copia o nomeAnterior e cola em nome para não ocorrer o bug descrito acima
        if(strcmp(num_Deletar,num_carteira) != 0)//compara num_Deletar com num_carteira se for diferente ele copiar para lista cadastro temp
        {
           fprintf(arquivoTemp, "%s %s %s %s %s %s %d %s %s\n", nome, ult_nome, telefone, cpf_sdigito, cpf_digito, rg, nascimento, email, num_carteira);
        }
        else//Se não apagou vira true
        {
           apagou = true;
        }
     }
  }
  FecharArquivo(arquivo);//Fecha arquivo
  FecharArquivo(arquivoTemp);//Fecha arquivo Temp
  arquivo = AbreArquivo('g', "ListaCadastro.txt");//Abre arquivo Lista Cadastro em modo de gravação
  arquivoTemp = AbreArquivo('l', "ListaCadastroTemp.txt");//Abre arquivo Lista Cadastro em modo de leitura

  strcpy(nomeAnterior,"");//Copia nomeAnterior para "NULL" para a variável ficar vazia

  while(!feof(arquivoTemp))//Usa o loop para ler o arquivo todo com o ponteiro *arquivoTemp
  {
     fscanf(arquivoTemp,"%s %s %s %s %s %s %d %s %s", &nome, &ult_nome, &telefone, &cpf_sdigito, &cpf_digito, &rg, &nascimento, &email, &num_carteira);//Escaneia tudo que estar em Lista Cadastro e joga nas variáveis
     if(strcmp(nomeAnterior,nome) != 0)//Se nomeAnterior for diferente de nome ele copiar todos os arquivos de volta para Lista Cadastro
     {
        strcpy(nomeAnterior,nome);//Copia o nomeAnterior e cola em nome para não ocorrer o bug que ira apararecer o ultimo conveniado 2 vezes
        fprintf(arquivo, "%s %s %s %s %s %s %d %s %s\n", nome, ult_nome, telefone, cpf_sdigito, cpf_digito, rg, nascimento, email, num_carteira);
     }
  }

  FecharArquivo(arquivo);//Fecha arquivo
  FecharArquivo(arquivoTemp);//Fecha arquivo Temp

  if(apagou)//Se apagou for False ele irá printar Conveniado deletado com sucesso
  {
     printf("Conveniado deletado com sucesso!\n\n");
```

```
    }
    else//Se apagou for true ele printa Conveniado não encontrado
    {
        printf("Conveniado não encontrado!\n\n");
    }
}


//função para cadastrar Atendimento
void CadastraUserGeral(char login[30], char senha[30], char permissao[30]);
void CadastraUserGeral(char login[30], char senha[30], char permissao[30])
{
    //Ponteiro
    FILE *arquivo;

    arquivo = AbreArquivo('a', "SistemadeLoginGeral.txt");//Abre o arquivo Sistema de
Login Geral em modo append
    fprintf(arquivo, "%s %s %s \n", login, senha, permissao); //printa as variáveis no
arquivo
    FecharArquivo(arquivo);//Fecha arquivo
}

//função para deletar o usuário na sessão de atendimento
void DeletarUserGeral(char deletarlogin[30]);
void DeletarUserGeral(char deletarlogin[30])
{

    setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
    //Ponteiros e Variáveis
    FILE *arquivo;
    FILE *arquivoTemp;
    char login [30];
    char senha[30];
    char permissao[30];
    char verificalogin[30];
    char verificasenha[30];
    char loginAnterior[30];
    bool apagou = false;

    arquivo = AbreArquivo('l', "SistemadeLoginGeral.txt");//Abre o arquivo Sistema de
Login Geral em modo de leitura
    arquivoTemp = AbreArquivo('g', "SistemadeLoginGeralTemp.txt");//Abre o arquivo
Sistema de Login Geral em modo de gravação

    while(!feof(arquivo))//Usa o loop para ler o arquivo todo com o ponteiro arquivo
    {
```

```
      fscanf(arquivo,"%s %s %s", &login, &senha, &permissao);//Escaneia tudo que
estar em Sistema de login Geral e joga nas variáveis
      // Zero = são iguais
      // Diferente de Zero = não são iguais
      if(strcmp(loginAnterior,login) != 0)
      {
         strcpy(loginAnterior,login);
         if(strcmp(deletarlogin,login) != 0)//Compara os deletarlogin com login, se for
diferente printa todos em Sistema de login Geral Temp
         {
            fprintf(arquivoTemp, "%s %s %s\n", login, senha, permissao);
         }
         else//Se não apagou vira True
         {
            apagou = true;
         }
      }
   }
   FecharArquivo(arquivo);//Fechar arquivo
   FecharArquivo(arquivoTemp);//Fechar arquivo Temp
   arquivo = AbreArquivo('g', "SistemadeLoginGeral.txt");//Abre o arquivo Sistema de
Login Geral em modo de gravação
   arquivoTemp = AbreArquivo('l', "SistemadeLoginGeralTemp.txt");//Abre o arquivo
Sistema de Login Geral Temp em modo de leitura

   strcpy(loginAnterior,"");//Copia loginAnterior para "NULL" para a variável ficar vazia

   while(!feof(arquivoTemp))//Usa o loop para ler o arquivo todo com o ponteiro
*arquivo Temp
   {
      fscanf(arquivoTemp,"%s %s %s", &login, &senha, &permissao);//Escaneia tudo
que estar em Lista Cadastro e joga nas variáveis
      if(strcmp(loginAnterior,login) != 0)//compara loginAnteriror com login se for
diferente ele printa em Sistema de Login Geral Temp
      {
         strcpy(loginAnterior,login);//copiar loginAnterior e cola em login para não
ocorrer o bug que apaga os 2 últimos logins da lista
         fprintf(arquivo, "%s %s %s\n", login, senha, permissao);
      }
   }

   FecharArquivo(arquivo);//Fecha arquivo
   FecharArquivo(arquivoTemp);//Fecha arquivo Temp

   if(apagou)//Se apagou for False ele irá printar usuário deletado com sucesso
   {
```

```
        printf("Usuário deletado com sucesso!\n\n");
    }
    else//Se apagou for True printar usuário não encontrado
    {
        printf("Usuario não encontrado!\n\n");

    }
}

//Função que lista todos os usuários no sistema
void ListarUserGeral();
void ListarUserGeral()
{
    setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
    //Ponteiro e Variáveis
    FILE *arquivo;
    char login[30];
    char senha[30];
    char permissao[30];
    char nomeAnterior[50];

    printf("LISTA DE USUARIOS\n");

    arquivo = AbreArquivo('l',"SistemadeLoginGeral.txt");//Abre o arquivo Sistema de
Login Geral em modo de leitura
    while(!feof(arquivo))//Usa o loop para ler o arquivo todo com o ponteiro arquivo
    {
        fscanf(arquivo,"%s %s %s", &login, &senha, &permissao);//Escaneia tudo que
estar em Sistema de login Geral e joga nas variáveis
        if(strcmp(nomeAnterior,login) != 0)//Compara o nomeAnterior com login para não
acontecer um bug que mostra o ultimo login, senha e permissão 2 vezes
        {
            strcpy(nomeAnterior,login);//copia nomeAnterior com login para não ocorrer o
bug descrito acima
            printf("\n|-----------------------------------------------------\n|Nome:..... %s
\n|Senha:..... %s \n|Permissão:..... %s", login, senha, permissao);//printa na
tela os logins com senha e permissão
        }
    }
    FecharArquivo(arquivo);//Fechar arquivo
}

//função para cadastrar consulta no faturamento
void CadastraRelatorioFaturamento(char nome[30], char ult_nome[30], char
num_carteira[30], int gasto_consulta, int dia_consulta, int mes_consulta, int
ano_consulta);
```

```
void CadastraRelatorioFaturamento(char nome[30], char ult_nome[30], char
num_carteira[30], int gasto_consulta, int dia_consulta, int mes_consulta, int
ano_consulta)
{

   //Ponteiro
   FILE *arquivo;

   arquivo = AbreArquivo('a', "Relatorio.txt");//Abre o arquivo Relatorio em modo
append
   fprintf(arquivo, "%s %s %s %d %d %d %d\n", nome, ult_nome, num_carteira,
gasto_consulta, dia_consulta, mes_consulta, ano_consulta); //printa as variáveis no
arquivo
   FecharArquivo(arquivo);//fecha arquivo

}

//função para listar o faturamento
void ListarRelatorioFaturamento();
void ListarRelatorioFaturamento()
{

   setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
   //Ponteiros e variáveis
   FILE *arquivo;
   char nome[50];
   char ult_nome[50];
   char numAnterior[30];
   char num_carteira[30];
   int gasto_consulta;
   int dia_consulta;
   int mes_consulta;
   int ano_consulta;

   printf("\nFaturamento Mensal: \n");

   arquivo = AbreArquivo('l',"Relatorio.txt");//Abre o arquivo Relatório em modo de
leitura
   while(!feof(arquivo))//Usa o loop para ler o arquivo todo com o ponteiro arquivo
   {
      fscanf(arquivo,"%s %s %s %d %d %d %d", &nome, &ult_nome, &num_carteira,
&gasto_consulta, &dia_consulta, &mes_consulta, &ano_consulta);//Escaneia tudo que
estar em Relatorio e joga nas variáveis
      printf("|Nome: \t\tNC: \t   Gastos do paciente:   data: \n|%s %s  |  %s |  R$:%d
| %d/%d/%d \n\n", nome, ult_nome, num_carteira, gasto_consulta, dia_consulta,
mes_consulta, ano_consulta);//printa nas telas as variáveis
```

```
    }
    FecharArquivo(arquivo);//fecha arquivo
}

//Pede pra o usuario escolher voltar pro menu ou fazer outra ação
int volta_menu_login();
int volta_menu_login()
{

    do
    {
        printf("Digite 0 para sair para o menu: ");//pergunta para o usuário se deseja
retornar ao menu de login
        scanf("%d", &volta_menu);

        if(volta_menu == 0)//Se o usuário digitar 0 ele retorna ao menu de login
        {
            system("cls");
            return main();
        }
        else//Se ele digitar outra coisa printa na tela opção inválida e faz um loop para
voltar ao começo da função
        {
            system("cls");
            printf("\nOpção Inválida\n\n");
            sleep(0,5);
        }
    }
    while(volta_menu != 0);
}

//Pede pra o usuário escolher voltar pro menu ou fazer outra ação
int volta_menu_atendimento();
int volta_menu_atendimento()
{

    do
    {
        printf("Digite 1 para retornar ao menu: ");//pergunta para o usuário se deseja
retornar ao menu de atendimento
        scanf("%d", &volta_menu);

        if(volta_menu == 1)//Se o usuário digitar 0 ele retorna ao menu de atendimento
        {
            return principal_atendente();
        }
```

```
        else
        {
            printf("\nOpção Invalida\n\n");//Se ele digitar outra coisa printa na tela opção
inválida e faz um loop para voltar ao começo da função
            sleep(0,5);
        }
    }
    while(volta_menu != 1);
}

//Pede para o usuario escolher voltar para o menu ou fazer outra ação
int volta_menu_medico();
int volta_menu_medico()
{

    do
    {
        printf("Digite 1 para retornar ao menu: ");//pergunta para o usuário se deseja
retornar ao menu de médico
        scanf("%d", &volta_menu);

        if(volta_menu == 1)
        {
            return principal_medico();//Se o usuário digitar 0 ele retorna ao menu de
médico
        }
        else//Se ele digitar outra coisa printa na tela opção inválida e faz um loop para
voltar ao começo da função
        {
            printf("\nOpção Invalida\n\n");
            sleep(0,5);
        }
    }
    while(volta_menu != 1);
}

//Pede para o usuário escolher voltar pro menu ou fazer outra ação
int volta_menu_conveniado();
int volta_menu_conveniado()
{

    do
    {
        printf("Digite 1 para retornar ao menu: ");//pergunta para o usuário se deseja
retornar ao menu de médico
        scanf("%d", &volta_menu);
```

```
        if(volta_menu == 1){//Se o usuário digitar 0 ele retorna ao menu de conveniado


            return principal_conveniado();
        }else//Se ele digitar outra coisa printa na tela opção inválida e faz um loop para
voltar ao começo da função
        {
            printf("\nOpção Invalida\n\n");
            sleep(0,5);
        }
    }while(volta_menu != 1);
}

//Pede pra o usuário escolher voltar pro menu ou fazer outra ação
int volta_menu_adm();
int volta_menu_adm()
{

    do
    {
        printf("\nDigite 1 para retornar ao menu: ");//pergunta para o usuário se deseja
retornar ao menu de administrador
        scanf("%d", &volta_menu);

        if(volta_menu == 1)//Se o usuário digitar 0 ele retorna ao menu de conveniado
        {
            return principal_adm();
        }
        else//Se ele digitar outra coisa printa na tela opção inválida e faz um loop para
voltar ao começo da função
        {
            printf("\nOpção Invalida\n\n");
            sleep(0,5);
        }
    }
    while(volta_menu != 1);
}

//função para logar Atendimento //pronta
void sistemadeloginGeral();
void sistemadeloginGeral()
{
    //Ponteiros e Variáveis
    FILE *arquivo;
    char login [30];
```

```
   char senha[30];
   char permissao[30];
   char verificalogin[30];
   char verificasenha[30];
   bool achou = false;//Coloca essa variável em FALSE para ter uma confirmação mais a frente

   do
   {
      printf("Digite seu login: ");//Aqui ele irá pedi o login e senha e adicionar nas variáveis
      setbuf(stdin,NULL);
      gets(verificalogin);
      printf("\nDigite sua senha: ");
      setbuf(stdin,NULL);
      gets(verificasenha);


      arquivo = AbreArquivo('l',"SistemadeLoginGeral.txt");//Abre o arquivo Sistema de login Geral em modo leitura
      while(!feof(arquivo))//Usa o loop para ler o arquivo todo com o ponteiro arquivo
      {
         fscanf(arquivo, "%s %s %s", &login, &senha, &permissao);//Escaneia todos os logins, senhas e permissões
         if(strcmp(verificalogin, login) == 0 && strcmp(verificasenha, senha) == 0)//compara o login, senha e permissões do arquivo com as digitadas pelo o usuário
         {
            achou = true;//se o login e senha forem iguais achou vira True
            if(strcmp(permissao, "Medico") == 0 )//Se a permissão que o usuário for igual a medico ele entrará no menu de médico
            {
               printf("Bem vindo Doutor (a).");
               sleep(2);
               system("cls");
               printf("Carregando...");
               sleep(1,5);
               system("cls");
               principal_medico();
            }

            if(strcmp(permissao, "Atendente") == 0 )//Se a permissão que o usuário for igual a atendente ele entrará no menu de atendimento
            {
               principal_atendente();
               printf("Bem vindo Atendente");
               sleep(2);
```

```
                system("cls");
                printf("Carregando...");
                sleep(1,5);
                system("cls");
            }

            if(strcmp(permissao, "Administrador") == 0 )//Se a permissão que o usuário
for igual a Administrador ele entrará no menu de administrador
            {
                printf("Bem vindo Administrador");
                sleep(2);
                system("cls");
                printf("Carregando...");
                sleep(1,5);
                system("cls");
                principal_adm();
            }

        }
        else//Se o login e a senha for diferente achou ira ser False e ele irá printa a
mensagem na tela
        {
            printf("Caso Esqueceu seu login ou senha entre em contato com o
administrador\nd4niel.arruda@gmail.com\n\nESPERE\n");
            sleep(2);
            system("cls");
        }
      }

   }
   while(!achou);//Depois irá dar o loop para repetir a função
   FecharArquivo(arquivo);//Fecha arquivo

}

//tela de menu do atendimento e suas funções
void principal_atendente();
void principal_atendente()
{

   setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
   //Variáveis
   int escolha_tela_atendente;
   char nome[30];
   char nome_medico[30];
   char ult_nome[30];
```

```
char telefone[20];
char cpf_sdigito[15];
char cpf_digito[15];
char rg[15];
int nascimento;
char email[40];
int dia_consulta;
int hora_consulta;
int minuto_consulta;
char especialidade[30];
char num_carteira[30];
char unidade[20];

//Menu do Atendimento
system("cls");
printf("\n\t\tMENU");
printf("\n");
printf("\t ----------------------------------------");
printf("\n\t| 1 - Cadastrar conveniado              |\n");
printf("\t|----------------------------------------");
printf("\n\t| 2 - Listar conveniados                |\n");
printf("\t|----------------------------------------");
printf("\n\t| 3 - Pesquisar conveniados             |\n");
printf("\t|----------------------------------------");
printf("\n\t| 4 - Deletar cadastro                  |\n");
printf("\t|----------------------------------------");
printf("\n\t| 5 - Agendar Consulta                  |\n");
printf("\t|----------------------------------------");
printf("\n\t| 6 - Listar Consultas                  |\n");
printf("\t|----------------------------------------");
printf("\n\t| 7 - Pesquisar Consultas               |\n");
printf("\t|----------------------------------------");
printf("\n\t| 8 - Deletar Consultas                 |\n");
printf("\t|----------------------------------------");
printf("\n\t| 9 - Situação dos feedbacks            |\n");
printf("\t|----------------------------------------");
printf("\n\t| 10 - Deletar feedbacks                |\n");
printf("\t|----------------------------------------");
printf("\n\t| 11 - Relatórios de faturamento  e tabela |\n");
printf("\t ----------------------------------------");
printf("\n\t| 0 - Sair                              |\n");
printf("\t ----------------------------------------");


printf("\nDigite uma opcao: ");//Pede para escolher um opção
sleep(0,5);
```

```
   sleep(0,5);//pausa a tela por x segundos
   scanf("%d", &escolha_tela_atendente);

   sleep(1);
   system("cls");//limpa a tela

   switch (escolha_tela_atendente)
   {
   case 1:
      //Cadastrar conveniado

      //Pede as informações para ser adicionada em Cadastra
      printf("\nDigite o nome: ");
      setbuf(stdin,NULL);//Sempre limpar o Buffer para receber uma nova entrada de
dados.
      gets(nome);
      printf("\nDigite o ultimo nome: ");
      setbuf(stdin,NULL);
      gets(ult_nome);
      printf("\nDigite o telefone: ");
      setbuf(stdin,NULL);
      gets(telefone);
      printf("\nDigite os 9 primeiros número do seu CPF sem ponto ou traço: ");
      setbuf(stdin,NULL);
      gets(cpf_sdigito);
      printf("\nDigite os últimos 2 números do seu CPF: ");
      setbuf(stdin,NULL);
      gets(cpf_digito);
      printf("\nDigite o RG: ");
      setbuf(stdin,NULL);
      gets(rg);
      printf("\nDigite o Nascimento: ");
      scanf("%d", &nascimento);
      printf("\nDigite o Email: ");
      setbuf(stdin,NULL);
      gets(email);
      printf("\nDigite o número da carteirinha: ");
      setbuf(stdin,NULL);
      gets(num_carteira);
      Cadastra(nome, ult_nome, telefone, cpf_sdigito, cpf_digito, rg, nascimento, email,
num_carteira);//Depois que coloca o que foi pedido nas veriáveis ele começa a função
cadastra
      volta_menu_atendimento();//Depois que ele terminar a função cadastra ele pode
escolher se vai voltar para o menu
      break;
```

```
    case 2:
        //Listar conveniados
        Listar();//Função para listar
        volta_menu_atendimento();
        break;

    case 3:
        //pesquisar conveniado
        ;//Para pode colocar um char dentro de um case precisar adicionar este ";"
        char pesquisa_nome[50];

        printf("\nDigite o nome do conveniado que deseja procurar: ");
        setbuf(stdin,NULL);
        gets(pesquisa_nome);
        PesquisaConveniado(pesquisa_nome);
        volta_menu_atendimento();
        break;

    case 4:
        //Deletar cadastro
        ;//Para pode colocar um char dentro de um case precisar adicionar este ";"
        char num_Deletar[50];

        printf("\nDigite o número de carteira do conveniado que deseja deletar: ");
        setbuf(stdin,NULL);
        gets(num_Deletar);
        Deletar(num_Deletar);
        volta_menu_atendimento();
        break;

    case 5:
        //cadastrar agenadamento
        //Pede as informações para ser adicionada em Agendar
        printf("\nDigite o nome: ");
        setbuf(stdin,NULL);//Sempre limpar o Buffer para receber uma nova entrada de
dados.
        gets(nome);
        printf("\nDigite o ultimo nome: ");
        setbuf(stdin,NULL);
        gets(ult_nome);
        printf("\nDigite o telefone: ");
        setbuf(stdin,NULL);
        gets(telefone);
        printf("\nDigite o nome do médico: ");
        setbuf(stdin,NULL);
        gets(nome_medico);
```

```
        printf("\nDigite os 9 primeiros número do seu CPF sem ponto ou traço: ");
        setbuf(stdin,NULL);
        gets(cpf_sdigito);
        printf("\nDigite os últimos 2 números do seu CPF: ");
        setbuf(stdin,NULL);
        gets(cpf_digito);
        printf("\nDigite o dia, mês e ano da consulta sem barra: ");
        scanf("%d", &dia_consulta);
        printf("\nDigite a hora da consulta: ");
        scanf("%d", &hora_consulta);
        printf("\nHorario: %d:00 \nDigite o minuto da consulta: ", hora_consulta);
        scanf("%d", &minuto_consulta);
        printf("\nDigite a especialidade da consulta: ");
        setbuf(stdin,NULL);
        gets(especialidade);
        printf("\nDigite o número da carteirinha: ");
        setbuf(stdin,NULL);
        gets(num_carteira);
        printf("\n| Paz | Cancioneiro | Paulista |");
        printf("\nDigite a o nome da unidade: ");
        setbuf(stdin,NULL);
        gets(unidade);
        Agendar(nome, ult_nome, nome_medico, telefone, cpf_sdigito, cpf_digito,
dia_consulta, hora_consulta, minuto_consulta, especialidade, num_carteira, unidade);
        volta_menu_atendimento();
        break;

    case 6:
        //listar agendamentos
        ListarAgenda();
        volta_menu_atendimento();
        break;

    case 7:
        //pesquisa agendamento
        ;//Para pode colocar um char dentro de um case precisar adicionar este ";"
        char pesquisa_num[30];

        printf("\nDigite o número de carteirinha do paciente para vê as consultas do
conveniado: ");
        setbuf(stdin,NULL);
        gets(pesquisa_num);
        PesquisaAgendamento(pesquisa_num);
        volta_menu_atendimento();
    case 8:
        //deletar agendamentos
```

```
        ;//Para pode colocar um char dentro de um case precisar adicionar este ";"
        char nomeDeletar[50];
        char nomeDeletarAgenda[50];
        char num_carteiraDeletar[30];
        int diaDeletar;
        int horarioDeletar;
        int minutoDeletar;

        printf("\nDigite um nome: ");
        setbuf(stdin,NULL);
        gets(nomeDeletar);
        printf("\nDigite o número de carteirinha: ");
        setbuf(stdin,NULL);
        gets(num_carteiraDeletar);
        printf("\nDigite o dia, mês e ano da consulta sem barra: ");
        scanf("%d", &diaDeletar);
        printf("\nDigite a hora da sua consulta: ");
        scanf("%d", &horarioDeletar);
        printf("\nHorario: %d:00 \nDigite o minuto da consulta: ", hora_consulta);
        scanf("%d", &minutoDeletar);
        DeletarAgendamento(nomeDeletarAgenda, num_carteiraDeletar, diaDeletar,
horarioDeletar, minutoDeletar);
        volta_menu_atendimento();
        break;

    case 9:
        //listar feedbacks
        ListarFeedback();
        volta_menu_atendimento();
        break;

    case 10:
        //excluir feedbacks
        ;//Para pode colocar um char dentro de um case precisar adicionar este ";"
        int excluir_feedback;

        printf("Para não haver manipulação dos feedbacks a função deletar feedback
excluir todos os feedbacks de uma vez.\n\n");
        printf("Se tive certeza disso digite 1 se não digite 0 para voltar ao menu: ");
        scanf("%d", &excluir_feedback);


        if(excluir_feedback == 1)//If para ter a certeza de exclusão dos feedbacks
        {
            DeletarFeedback();
        }
```

```
        break;

    case 11:
        ;
        //menu com relatórios
        int escolha_menu_relatorio;
        //Menu de relatórios e tabela
        printf("\n\t\Relatórios de faturamento");
        printf("\n");
        printf("\t ----------------------------------------");
        printf("\n\t| 1 - Adicionar faturamento no relatório    |\n");
        printf("\t|----------------------------------------");
        printf("\n\t| 2 - Listar Faturamento                    |\n");
        printf("\t|----------------------------------------");
        printf("\n\t| 3 - Tabela de Preço de Consultas e Exames |\n");
        printf("\t|----------------------------------------");
        printf("\n\t| 0 - Sair                                  |\n");
        printf("\t|----------------------------------------");

        printf("\nDigite uma opção: ");
        sleep(0,5);
        sleep(0,5);//pausa a tela por x segundos
        scanf("%d", &escolha_menu_relatorio);

        sleep(1);
        system("cls");//limpa a tela

        switch (escolha_menu_relatorio)
        {
        case 1:
            //Cadastrar relatório
            ;//Para pode colocar um char dentro de um case precisar adicionar este ";"

            char nome[30];
            char ult_nome[30];
            char num_carteira[30];
            int gasto_consulta;
            int dia_consulta;
            int mes_consulta;
            int ano_consulta;

            printf("\nDigite o nome do conveniado: ");
            setbuf(stdin,NULL);//Sempre limpar o Buffer para receber uma nova entrada de
dados.
            gets(nome);
            printf("\nDigite o último nome do conveniado: ");
```

```
            setbuf(stdin,NULL);
            gets(ult_nome);
            printf("\nDigite o número da carteirinha: ");
            setbuf(stdin,NULL);
            gets(num_carteira);
            printf("\nDigite o gasto da consulta hoje: ");
            scanf("%d", &gasto_consulta);
            printf("\nDigite o dia da consulta: ");
            scanf("%d", &dia_consulta);
            printf("\nDigite o mês da consulta: ");
            scanf("%d", &mes_consulta);
            printf("\nDigite o ano da consulta: ");
            scanf("%d", &ano_consulta);
            CadastraRelatorioFaturamento(nome, ult_nome, num_carteira, gasto_consulta,
dia_consulta, mes_consulta, ano_consulta);
            volta_menu_atendimento();
            break;

        case 2:
            //Listar Relatório
            ListarRelatorioFaturamento();//Lista Relatório
            volta_menu_atendimento();
            break;

        case 3:
            printf("TABELA DE CONSULTAS E EXAMES\n\n");
            printf("|----------------------------------------------------------------\n");
            printf("|Consulta Com Clinico Geral:..... R$ 50,00\n\n");
            printf("|Consulta Com Especialistas:..... R$ 90,00\n\n");
            printf("|Hemograma Completo:..... R$ 75,00\n\n");
            printf("|Raio X:..... R$ 40,00\n\n");
            printf("|Raio X Panorâmico:..... R$ 40,00\n\n");
            printf("|Ressonância Magnética Mama:..... R$ 300,00\n\n");
            printf("|Ressonância Magnética Próstata:..... R$ 320,00\n\n");
            printf("|Ultrasson Mamas:..... R$ 55,00\n\n");
            printf("|Ultrasson Próstata Abdominal:..... R$ 55,00\n\n");
            printf("|Ultrasson Próstata Transretal:..... R$ 65,00\n\n");
            printf("|Urografia Venosa Excretora:..... R$ 160,00\n\n");
            printf("|Urografia Venosa Minutada:..... R$ 220,00\n");
            printf("|----------------------------------------------------------------\n");

            volta_menu_atendimento();
            break;
        case 0:
            //volta ao menu de Atendimento
            volta_menu_atendimento();
```

```
            break;

        default:
            printf("Opção incorreta\n");
            sleep(1,25);
            volta_menu_atendimento();
            break;
        }
        break;

    case 0:
        //volta ao menu de login
        volta_menu_login();
        break;

    default:
        system("cls");
        printf("\n\Opção invalida! Tente Novamente!\n\n");
        sleep(1,5);
        volta_menu_atendimento();
        break;
    }



}

//tela de menu do conveniado e suas funções
void principal_conveniado();
void principal_conveniado()
{

    setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
    //Variáveis
    int escolha_tela_conveniado;
    char nome[30];
    char nome_medico[30];
    char ult_nome[30];
    char telefone[20];
    char cpf_sdigito[15];
    char cpf_digito[15];
    int dia_consulta;
    int hora_consulta;
    int minuto_consulta;
    char especialidade[30];
    char num_carteira[30];
```

```
    char unidade[20];

    //Menu de conveniado
    system("cls");
    printf("\n\t\tMENU");
    printf("\n");

    printf("\t|----------------------------");
    printf("\n\t| 1 -  Agendar Consulta      |\n");
    printf("\t|----------------------------");
    printf("\n\t| 2 - Listar Suas Consultas  |\n");
    printf("\t|----------------------------");
    printf("\n\t| 3 - Desmarcar Suas consultas|\n");
    printf("\t|----------------------------");
    printf("\n\t| 4 - Dar um feedback        |\n");
    printf("\t|----------------------------");
    printf("\n\t| 5 - Nossos Endereços       |\n");
    printf("\t| ----------------------------");
    printf("\n\t| 0 - Sair                   |\n");
    printf("\t|----------------------------");

    printf("\nDigite uma opção: ");
    sleep(0,5);
    sleep(0,5);//pausa a tela por x segundos
    scanf("%d", &escolha_tela_conveniado);

    sleep(1);
    system("cls");//limpa a tela
    //Switch para escolha de opções no menu
    switch(escolha_tela_conveniado)
    {
    case 1:
       //cadastrar agendamento
       //Pede as informações para ser adicionada em Agendar
       printf("\nDigite seu primeiro nome: ");
       setbuf(stdin,NULL);//Sempre limpar o Buffer para receber uma nova entrada de
dados.
       gets(nome);
       printf("\nDigite seu último nome: ");
       setbuf(stdin,NULL);
       gets(ult_nome);
       printf("\nDigite seu telefone: ");
       setbuf(stdin,NULL);
       gets(telefone);
       printf("\nDigite o nome do médico: ");
       setbuf(stdin,NULL);
```

```
        gets(nome_medico);
        printf("\nDigite os 9 primeiros número do seu CPF sem ponto ou traço: ");
        setbuf(stdin,NULL);
        gets(cpf_sdigito);
        printf("\nDigite os últimos 2 números do seu CPF: ");
        setbuf(stdin,NULL);
        gets(cpf_digito);
        printf("\nDigite o dia, mês e ano que você deseja marcar a consulta sem barra: ");
        scanf("%d", &dia_consulta);
        printf("\nDigite o horario que você deseja para a sua consulta: ");
        scanf("%d", &hora_consulta);
        printf("\nHorario: %d:00 \nDigite o minuto que você deseja para a sua consulta: ",
hora_consulta);
        scanf("%d", &minuto_consulta);
        printf("\nDigite a especialidade da consulta: ");
        setbuf(stdin,NULL);
        gets(especialidade);
        printf("\nDigite o número da carteirinha: ");
        setbuf(stdin,NULL);
        gets(num_carteira);
        printf("\n| Paz | Cancioneiro | Paulista |");
        printf("\nDigite a o nome da unidade: ");
        setbuf(stdin,NULL);
        gets(unidade);
        Agendar(nome, ult_nome, nome_medico, telefone, cpf_sdigito, cpf_digito,
dia_consulta, hora_consulta, minuto_consulta, especialidade, num_carteira, unidade);
        volta_menu_conveniado();
        break;

    case 2:
        //pesquisa agendamento
        ;//Para pode colocar um char dentro de um case precisar adicionar este ";"
        char pesquisa_num[30];

        printf("\nDigite o número de carteirinha para ver suas consultas: ");
        setbuf(stdin,NULL);
        gets(pesquisa_num);
        PesquisaAgendamento(pesquisa_num);
        volta_menu_conveniado();
        break;

    case 3:
        //deletar agendamentos
        ;//Para pode colocar um char dentro de um case precisar adicionar este ";"
        char nomeDeletar[50];
        char nomeDeletarAgenda[50];
```

```
        char num_carteiraDeletar[30];
        int diaDeletar;
        int horarioDeletar;
        int minutoDeletar;

        printf("\nDigite um nome: ");
        setbuf(stdin,NULL);
        gets(nomeDeletar);
        printf("\nDigite o número de carteirinha: ");
        setbuf(stdin,NULL);
        gets(num_carteiraDeletar);
        printf("\nDigite o dia, mês e ano da consulta sem barra: ");
        scanf("%d", &diaDeletar);
        printf("\nDigite o hora da sua consulta: ");
        scanf("%d", &horarioDeletar);
        printf("\nHorario: %d:00 \nDigite o minuto da consulta: ", horarioDeletar);
        scanf("%d", &minutoDeletar);
        DeletarAgendamento(nomeDeletarAgenda, num_carteiraDeletar, diaDeletar,
horarioDeletar, minutoDeletar);
        volta_menu_conveniado();
        break;

    case 4:
        ;//Para pode colocar um char dentro de um case precisar adicionar este ";"
        char nome_medico[30];
        int nota_medico;
        int nota_atendimento;

        printf("\nDigite o nome do médico");
        printf("(Andre) (Osvaldo) (Luis): ");
        setbuf(stdin,NULL);
        gets(nome_medico);
        printf("\nDigite uma nota de 0 a 10 para o atendimento de seu médico: ");
        scanf("%d", &nota_medico);
        printf("\nDigite uma nota de 0 a 10 para seu atendimento em geral: ");
        scanf("%d", &nota_atendimento);
        printf("Muito Obrigado pelo seu Feedback :)\n\n");
        CadastraFeedback(nome_medico, nota_medico, nota_atendimento);
        volta_menu_conveniado();

        break;

    case 5:
        //Aqui mostra o endereço das clínicas e o contato das mesmas
        printf("Onde você pode nos encontrar :)\n");
```

```
        printf("\n|-----------------------------------------------------------------------------------------|\n");
        printf("|   CLINICA PRINCIPAL: R. da Paz, 797 - Chácara Santo Antônio, São Paulo
- SP, 04713-000\n");
        printf("|   Tel.: (11) 5181-4055 \n");
        printf("|-----------------------------------------------------------------------------------------|\n");

        printf("|   2° CLINICA: Rua Cancioneiro Popular, 210 - Chácara Santo Antônio São
Paulo - SP, 04710-000\n");
        printf("|   Tel.: (11) 2114-4000 \n");
        printf("|-----------------------------------------------------------------------------------------|\n");

        printf("|   Av. Paulista, 900 - Cerqueira César São Paulo - SP CEP 01310-100\n");
        printf("|   Tel.: (11) 3170-3700 \n");
        printf("|-----------------------------------------------------------------------------------------|\n\n");

        volta_menu_conveniado();
        break;

    case 0:
        volta_menu_login();
        break;

    default:
        system("cls");
        printf("\n\Opção invalida! Tente Novamente! \n\n");
        sleep(1,5);
        volta_menu_conveniado();
        break;
    }
}

//tela de menu do administrador e suas funções
void principal_adm();
void principal_adm()
{

    setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
    char login[30];
    char senha[30];
    char permissao[30];
    int escolha_tela_adm;

    //Menu de Administrador do sistema
    system("cls");
    printf("\n\t\tMENU");
    printf("\n");
```

```
    printf("\t|----------------------------");
    printf("\n\t| 1 - Adicionar Usuários      |\n");
    printf("\t|----------------------------");
    printf("\n\t| 2 - Listar Usuários         |\n");
    printf("\t|----------------------------");
    printf("\n\t| 3 - Deletar Usuários        |\n");
    printf("\t ----------------------------");
    printf("\n\t| 0 - Sair                    |\n");
    printf("\t ----------------------------");

        printf("\n");
        printf("\nDigite uma opção: ");
    sleep(0,5);
    sleep(0,5);//pausa a tela por x segundos
    scanf("%d", &escolha_tela_adm);

    sleep(1);
    system("cls");//limpa a tela
    //Switch de escolha de opções no menu de Administrador
    switch (escolha_tela_adm)
    {
    case 1:
       //Pede as informações para ser adicionada como um novo usuário
       printf("Digite o nome do novo usuário: ");
       setbuf(stdin,NULL);//Sempre limpar o Buffer para receber uma nova entrada de
dados.
       gets(login);
       printf("\nDigite a Senha do novo usuário: ");
       setbuf(stdin,NULL);
       gets(senha);
       printf("\nDigite o tipo de Permissão que desejar dar ao usuário");
       printf("\n(Administador) (Medico) (Atendimento) (Conveniado)");
       printf(": ");
       setbuf(stdin,NULL);
       gets(permissao);
       CadastraUserGeral(login, senha, permissao);
       volta_menu_adm();
       break;

    case 2:
       //Lista todos os usuários
       ListarUserGeral();
       volta_menu_adm();
       break;
```

```
    case 3:
        //Deleta usuários
        ;//Para pode colocar um char dentro de um case precisar adicionar este ";"
        char deletarlogin[30];
        printf("\nDigite o nome do usuário que deseja deletar: ");
        setbuf(stdin,NULL);//Sempre limpar o Buffer para receber uma nova entrada de
dados.
        gets(deletarlogin);
        DeletarUserGeral(deletarlogin);
        volta_menu_adm();
        break;

    case 0:
        volta_menu_login();
        break;

    default:
        system("cls");
        printf("\n\nOpção invalida! Tente Novamente! \n\n");
        sleep(1,5);
        volta_menu_adm();
        break;
    }
}

//tela de menu do médico e suas funções
void principal_medico();
void principal_medico()
{

    setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR
    int escolha_tela_medico;

    system("cls");
    printf("\n\t\tMENU");
    printf("\n");
    //Menu de Médico
    printf("\t|--------------------------------------");
    printf("\n\t| 1 - Pesquisar Agendamento          |\n");
    printf("\t|--------------------------------------");
    printf("\n\t| 2 - Listar conveniados             |\n");
    printf("\t| --------------------------------------");
    printf("\n\t| 3 - Pesquisar conveniado           |\n");
    printf("\t|--------------------------------------");
    printf("\n\t| 0 - Sair                           |\n");
    printf("\t| --------------------------------------\n");
```

```
    printf("\nDigite uma opção: ");
    sleep(0,5);
    sleep(0,5);//pausa a tela por x segundos
    scanf("%d", &escolha_tela_medico);

    sleep(1);
    system("cls");//limpa a tela
    //Switch para escolha de opções no menu Médico
    switch (escolha_tela_medico)
    {
    case 1:
        ;//Para pode colocar um char dentro de um case precisar adicionar este ";"
        char pesquisa_num[30];

        printf("\nDigite o número de carteirinha do paciente para ver suas consultas: ");
        setbuf(stdin,NULL);//Sempre limpar o Buffer para receber uma nova entrada de
dados.
        gets(pesquisa_num);
        PesquisaAgendamento(pesquisa_num);
        volta_menu_medico();
        break;

    case 2:
        //Listar conveniados
        Listar();
        volta_menu_medico();
        break;

    case 3:
        //pesquisar conveniado
        ;
        char pesquisa_nome[50];

        printf("\nDigite o nome do conveniado que deseja procurar: ");
        setbuf(stdin,NULL);
        gets(pesquisa_nome);
        PesquisaConveniado(pesquisa_nome);
        volta_menu_medico();
        break;

    case 0:
        volta_menu_login();
        break;

    default:
```

```
        system("cls");
        printf("\nOpção incorreta\n");
        sleep (1,2);
        principal_medico();
        break;
    }




}


int main ()
{
    setlocale(LC_ALL, "Portuguese");//Coloca acento na linguagem PTBR

    int login_escolha; // variável usada no int main para escolher usuário
    int login_menu;
    //No int main deixamos apenas a tela de login para uma melhor organização do
código em si
    printf("\n");
    printf("\t\t\t\t\t\t\t*!Bem vindo!*\n\n");
    printf("\tDATA : %s\n\n\n",__DATE__);//Para aparecer a data perto do bem vindo
    sleep(1);

    printf("\n\t\t\t\t\t\t  | Usuários | \t\t");

printf("\n\t\t_____
_____\n\n");
    printf("\t\t|       0-Sair \t\t\t  1-Conveniado \t\t\t2-Usuários       |\n");

printf("\t\t_____
_____\n\n");

    printf("\n\tEscolha um opção por favor --> ");

    sleep(0,5);
    sleep(0,5);//pausa a tela por x segundos
    scanf("%d", &login_escolha);
    sleep(1);
    system("cls");//limpa a tela

    //Switch para escolha de login
    switch(login_escolha)
    {
```

```
    case 0:
        return 0;
        break;

    case 1:
        printf("Bem vindo Conveniado");
        sleep(2);
        system("cls");
        printf("Carregando...");
        sleep(2);
        system("cls");
        principal_conveniado();

    case 2:
        sistemadeloginGeral();
        break;
    default:
        printf("opção incorreta");
        sleep(1,25);
        system("cls");
    }
    return 0;
}
```