UNIVERSIDADE PAULISTA

GUILHERME DOS SANTOS

JOÃO VITOR VIZU

JOSÉ HENRIQUE ALVES DE OLIVEIRA

MICHAEL MAYER DE ASSIS

VICTOR HUGO RODRIGUES BERZOTTI

VITOR SOUSA ROSA

PROJETO INTEGRADO MULTIDISCIPLINAR II:

iMerc

RIBEIRÃO PRETO

2018

GUILHERME DOS SANTOS
JOÃO VITOR VIZU
JOSÉ HENRIQUE ALVES DE OLIVEIRA
MICHAEL MAYER DE ASSIS
VICTOR HUGO RODRIGUES BERZOTTI
VITOR SOUSA ROSA

PROJETO INTEGRADO MULTIDISCIPLINAR II:

iMerc

Projeto Integrado Multidisciplinar elaborado como parte das exigências para conclusão do semestre do curso de Análise e Desenvolvimento de Sistemas pela Universidade Paulista.

Orientador: Professor Mestre Marcelo Gomes de Paula

RIBEIRÃO PRETO
2018

GUILHERME DOS SANTOS
JOÃO VITOR VIZU
JOSÉ HENRIQUE ALVES DE OLIVEIRA
MICHAEL MAYER DE ASSIS
VICTOR HUGO RODRIGUES BERZOTTI
VITOR SOUSA ROSA

PROJETO INTEGRADO MULTIDISCIPLINAR I:

iMerc

Projeto Integrado Multidisciplinar elaborado como parte das exigências para conclusão do semestre do curso de Análise e Desenvolvimento de Sistemas pela Universidade Paulista - UNIP.

Orientador: Professor Mestre Marcelo Gomes de Paula

Aprovado em:

Universidade Paulista – UNIP
_____/__/___
Prof. Marcelo Gomes
Universidade Paulista – UNIP
_____/__/___
Universidade Paulista – UNIP
BANCA EXAMINADORA
_____/__/___

AGRADECIMENTOS

Agradecemos a Deus por nossas conquistas até aqui e por nos ter dado forças para superar as dificuldades. Aos nossos familiares que nos tem apoiado imensamente em todas as nossas escolhas. A nosso orientador, Professor Marcelo Gomes de Paula e todos professores, por toda dedicação e suporte proporcionados para que fizéssemos o melhor, bem como pelas correções e incentivos. A todos que direta ou indiretamente contribuíram para tornar possível a realização deste trabalho, nosso muito obrigado.

DEDICATÓRIA

Dedicamos este trabalho aos nossos familiares, amigos e demais pessoas que nos deram apoio, necessários para realização deste trabalho. A vocês nossa gratidão, não somente pela compreensão, mas principalmente por tornarem nossas vidas mais felizes.

RESUMO

O mercado Bom Preço encontra dificuldades em sua logística gerenciando o seu estoque inadequadamente permitindo que os mesmos estraguem ou faltem. O caixa contém problema em seu fechamento, ou seja, tem problemas em seu faturamento, envolvidos pelo atraso no atendimento devido a quantidade de produtos que o funcionário deve dar baixa. O sistema denominado iMerc foi desenvolvido pela empresa 2VJMIG para corresponder aos requisitos e expectativas do mercado Bom Preço ao informatizar seu estabelecimento. Um dos objetivos do Bom Preço para requisitar o software é acelerar o fluxo de caixa do seu mercado, mas a 2VJMIG vai além, permitindo que com o sistema iMerc possa gerenciar o cadastro de produtos, fornecedores e clientes, tendo controle sobre os mesmos. Para isso, o sistema foi implementado com menus específicos de vendas, cadastros, controle de estoque e controle de vendas. O sistema, conta com um conjunto de funções dando suporte ao o usuário de forma a auxilia-lo em suas atividades. A 2VJMIG disponibiliza aos seus clientes updates e upgrades, instalação de redes, implantação do software e suporte para manutenção dos mesmos. O presente trabalho foi desenvolvido para expor os conhecimentos adquiridos nas matérias de Lógica de Programação, Engenharia de Software, Redes de Computadores, Matemática e Ética, descritos em um sistema para mercado.

Palavras chave: Sistemas; Mercado; Controle; Gerenciamento.

ABSTRACT

The Bom Preço market is in difficulties in its management by managing its stock improperly allowing them to spoil or lack. The box contains the problem in your closing, that is, you have problems in your billing, due to the delay due to a quantity of products that the user must have low. The system called iMerc was developed by the company 2VJMIG to correspond to the requirements and expectations of the Bom Preço market when computerizing its establishment. One of the objectives of the Bom Preço to order the software is to accelerate the cash flow of its market, but 2VJMIG goes beyond that, allowing the iMerc system. The largest database of suppliers, suppliers and customers control control over the themselves. For this, the system was implemented with specific menus of sales, registrations, inventory control and sales control. The system counts on a set of functions giving support to the user in an auxiliary way in its activities. 2VJMIG provides its customers with updates and upgrades, network installation, software implementation and maintenance support. The present work was developed to expose the acquired knowledge in the areas of Programming Logic, Software Engineering, Computer Networks, Mathematics and Ethics, to index in a market system.

Keywords: Systems; Market; Control; Management.

LISTA DE ILUSTRAÇÕES

LISTA DE GRÁFICOS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Os supermercados e mercados em geral são importantes para a economia do país, segundo João Lourenço Couto Ferreira Júnior (2012) esse setor de mercado apresenta uma grande qualidade e inovação, sendo assim abrem filiais em diversos locais do país e fazem com que seja preciso a contratação de funcionários e melhor atenda a demanda da população.

Outro setor que vem crescendo é a tecnologia, e a tecnologia sempre está evoluindo e ajudando empresas, microempresas, mercados, supermercados, hipermercados e muitos outros setores do mercado a crescerem e expandirem seus horizontes.

O Comitê Gestor da Internet no Brasil investigou se a apropriação das ferramentas de tecnologia da informação e comunicação pelas organizações brasileiras está ocorrendo com finalidade estratégica, ou seja, se o uso da TIC (tecnologia da informação e comunicação) está potencializando novas práticas de gestão e o desenvolvimento de novos produtos e serviços para melhoria do desempenho organizacional de acordo com uma pesquisa de mercado da SEBRAE.

Nas empresas há a preocupação em aprimorar processos e atividades gerenciais, com foco em benefícios como a redução dos custos de transação e a melhoria da produtividade e crescimento das empresas como visto na pesquisa, sendo assim o mercado popularmente chamado de Bom Preço tende a utilizar tais apropriações.

Para solicitar tais avanços tecnológicos o mercado Bom Preço conversa com a empresa 2VJMIG, conceituada no ramo de desenvolvimento, onde solicita um software para facilitar a administração, o gerenciamento, e o fluxo de caixa do seu mercado.

Não somente procurando melhorar esses aspectos, o Bom Preço se viu necessário de utilizar meios que permite acesso fácil e ágil aos seus consumidores, onde que com tal estratégia consiga obter mais clientes para o seu estabelecimento.

Esse fácil acesso almeja permitir todos os tipos de consumidor, desde pessoas comuns até idosos, gestantes e deficientes, pois notaram que o número de portadores de deficiência vem crescendo no Brasil, e sendo crescente mostra que essa situação possui uma ótima inclusão social dando visibilidade ao mercado e resultando em uma vantagem para obter novos clientes.

Como visto a importância com a acessibilidade, o Bom Preço também se preocupará com o meio-ambiente visando utilizar estruturas para melhorar o mesmo, pois fazendo tais coisas, ele se vê mais próximo de atingir um de seus objetivos e possa vir a ser uma grande potência no ramo de varejo.

1.1 Objetivo

O objetivo do presente trabalho é construir o software iMerc, visando aplicar os conhecimentos adquiridos em Lógica de Programação, Engenharia de Software, Redes de Computadores, Matemática e Ética, indicar ao mercado Bom Preço soluções para aplicar a sustentabilidade e a acessibilidade, pois se nota que a tecnologia abrange várias áreas.

2 ACESSIBILIDADE

Visto que uma pessoa portadora de deficiência física, quando vai ao mercado é um consumidor, e tem os direitos de consumidor aplicados da mesma forma que pessoas não portadoras de deficiência, o mercado pode proporcionar acessibilidade e facilidade para que ele possa realizar suas compras normalmente como um consumidor comum. Começando com o carrinho de compras para deficientes, onde o carrinho tem um assento para que uma mãe caso tenha um filho de colo e ou deficiente possa colocá-lo ali, oferecendo assim uma acessibilidade e facilidade.

Para a locomoção de deficientes visuais pode-se colocar no chão pisos táteis, assim o cliente pode se locomover normalmente no mercado utilizando sua bengala. O mercado pode utiliza carrinhos especiais para os clientes que tenham ausência da visão, colocando um botão neste carrinho que caso haja uma situação que o cliente se sinta em pânico ele apertará o botão e um atendente irá socorrê-lo.

Nas gôndolas do mercado pode conter um botão que é utilizado por pessoas que têm deficiência visual, informando o valor do produto e seu respectivo nome. Como é visto no mercado, as gôndolas são altas, logo pode haver funcionários ficando à disposição caso um cliente que possua nanismo queira pegar algo na prateleira mais alta. Esses funcionários podem auxiliar também as pessoas idosas caso queiram pegar algum produto pesado como saco de arroz, caixa de leite ou até mesmo um produto que esteja em uma prateleira mais alta.

Fazendo assim, o mercado fornecerá acessibilidade aos portadores de alguma deficiência permitindo que sintam uma certa autonomia como consumidores.

3 SUSTENTABILIDADE

A sustentabilidade é importante pois é ela que faz um equilíbrio entre a sociedade e a natureza. Pensando em usar os recursos naturais para ajudar o meio ambiente o mercado pode gerar um sistema de Captação de Energia solar, e essa energia pode ser usada no mercado. Seguindo a ideia, um sistema de captação de água da chuva seria viável onde que a água captada é utilizada para a limpeza e higienização do chão do mercado, fazendo com que não seja gasto uma grande quantidade de água limpa para este fim.

O mercado pode organizar o recolhimento de baterias e pilhas que estão desgastadas ou perderam seu uso, esses materiais recolhidos são devidamente encaminhados para uma empresa responsável, e todo cliente que contribuir ganhará um desconto ou algo relativo, servindo de incentivo para ele contribuir com a natureza.

4 REVISÃO BIBLIOGRÁFICA

É visível que hoje em dia as empresas que conseguem se destacar, até mesmo as menores, são aquelas que estão em uma constante procura de aprimoramentos para sua estrutura e tentando inovar na área comercial em que atuam. Segundo o site “Administradores (2017)”, afirma que a falta de investimento de tecnologia por parte das empresas é um dos dez motivos que leva a falência prematura da mesma.

Implementar um sistema que pode gerenciar um mercado, onde o usuário tem ao seu dispor o controle completo do seu estabelecimento e de grande importância pra o varejista, tendo em mente o pensamento que:

> “O advento dessas novas tecnologias, impacta o dia a dia das pessoas a todo momento e, consequentemente, altera a forma como elas consomem. Se antes a preocupação dos lojistas era vender, hoje encaramos um desafio muito maior, que é oferecer a melhor experiência de compra. (Anselmo Martini, 2017, Administradores)”.

Há vários benefícios que o varejista deve levar em conta ao se implementar um sistema de gerenciamento de mercados, tais como:

Controle e segurança - Por lidar com um grande público, o varejista precisa implementar ferramentas para controlar a movimentação de produtos em sua loja. Por isso, é fundamental a automação comercial, uma vez que o caixa, ou ponto de venda, é de onde a mercadoria sai, e é justamente nesse local que acontecem erros e fraudes.

Redução de custos - Usando um software de gestão no supermercado será possível enxergar melhor as despesas de cada departamento de loja, e assim implementar ações para reduzi-las, evitando a evolução dessas despesas. Na gestão de estoque por exemplo, que é a base da operação do supermercadista e, ao mesmo tempo, um desafio manter seu controle correto, é possível controlar as perdas de mercadorias, e reduzir custos usando um software de gestão.

Aumento de produtividade - Quando você conta com um software de gestão no supermercado, uma série de processos são automatizados e integrados, reduzindo retrabalho e aumentando a produtividade da sua equipe. Por exemplo, ao dar entrada na nota fiscal, ela já alimenta o estoque e já fica registrada para realizar o cálculo de impostos e enviar as obrigações fiscais. Sem sistema, você tem de fazer cada passo separadamente.

Melhoria de atendimento ao cliente - Quando o cliente chega ao supermercado ele deseja ter uma ótima experiência de compra, dessa forma o varejista precisa evitar os atritos que podem prejudicar essa boa experiência do cliente. Cliente satisfeito vai passar muito tempo na sua loja,

já a insatisfação do cliente começa quando ele precisa ficar muito tempo na fila para pagar sua compra.

É de se observar que o varejista deve atender as demandas de seu cliente, com o uso da tecnologia varejistas podem ter a informatização como sua aliada, dessa forma fazendo com que seu comércio seja destaque, tornando–o mais atrativo:

> "A tecnologia se torna uma grande aliada, as lojas físicas se tornam cada vez mais atraentes, principalmente ao utilizar soluções interativas que mesclam vendas e entretenimento, proporcionando uma melhora significativa na experiência de compra. (Anselmo Martini, 2017, Administradores)"

Existem várias empresas que já estão a um bom tempo no ramo de informatização do varejo, como a empresa Cielo que apresenta a máquina "Lio", que recebe aplicativos produzidos pela própria Cielo ou por qualquer desenvolvedor para que o varejista tenha um computador inteiro em sua mão capaz de fazer toda operação que precisar, de maneira digital, da gestão do estoque ao registro de pedidos e recebimentos e pagamentos. A empresa Tyco, que conseguiu criar as prateleiras inteligentes, que são capazes de contar os itens que são retirados pelos consumidores, ele utiliza um sistema de sensores nas prateleiras que possibilita que o setor interno da loja saiba, em tempo real, quais mercadorias estão sendo retiradas, as informações vão para a nuvem e, organizadas, fornecem informações para reposição de estoque e contagem precisa e em tempo real de itens escolhidos pelos consumidores, além de manter os produtos organizados. Isso acontece pelo sistema de pusher, que empurra os itens para frente, evitando espaços vazios à mostra.

Tendo em vista isso, nossa equipe 2VJMG, decidiu realizar a construção de um software de gerenciamento para mercados.

4.1 Pesquisa de mercado

Tendo em vista o objetivo de criar um software que ofereça tais requisitos e funcionalidades, foi realizado um estudo para identificar o uso de softwares nas empresas onde que com tais informações disponibilizadas pela SEBRAE, possa ser feito um software de qualidade.

Obteve-se as seguintes informações em que um terço das empresas brasileiras introduziram softwares novos ou realizaram algum aperfeiçoamento significativo nos existentes. Postura que tem relação direta com o porte da empresa como visto na Tabela 1- Introdução de Software nas Empresas.

Tabela 1-Introdução de Software nas Empresas

Porte Empresa	Introdução de softwares
Pequena	25%
Media	42%
Grande	57%

Fonte: elaborado pelos autores.

Boa parte dos softwares novos que foram introduzidos pelas empresas corresponde a sistemas integrados de gestão empresarial, como os ERP (Sistemas Integrados de Gestão Empresarial e, em inglês, Enterprise Resource Planning). Também se destacaram programas que viabilizam os usos básicos do computador e da Internet, como os pacotes de softwares de edição de texto, de imagem, de planilhas eletrônicas e de antivírus;

verifica-se que quanto maior o porte da empresa maior é a incidência de uso de um software;

A proporção do uso de softwares adquiridos por encomenda, aqueles que foram desenvolvidos de forma especifica para uma instituição especializada, é maior nas empresas médias e grandes como visto a Tabela 2- Softwares sob Encomenda.

Tabela 2- Softwares sob Encomenda

Porte Empresa	Software sob Encomenda
Pequena	41%
Media	48%
Grande	56%

Fonte: elaborado pelos autores.

Para as empresas que declararam ter introduzido softwares novos ou que passaram por algum aperfeiçoamento significativo, o elemento motivador mais citado foi a melhoria de processos e procedimentos internos. O segundo motivo mais citado foi o ganho de produtividade e eficiência, outro motivo apontado refere-se às adequações por exigência da lei, como adoção de ponto eletrônico, notas fiscais eletrônicas ou registros contábeis informatizados como visto na Tabela 3 - Motivos Mencionados pelas Empresas.

Tabela 3 - Motivos Mencionados pelas Empresas

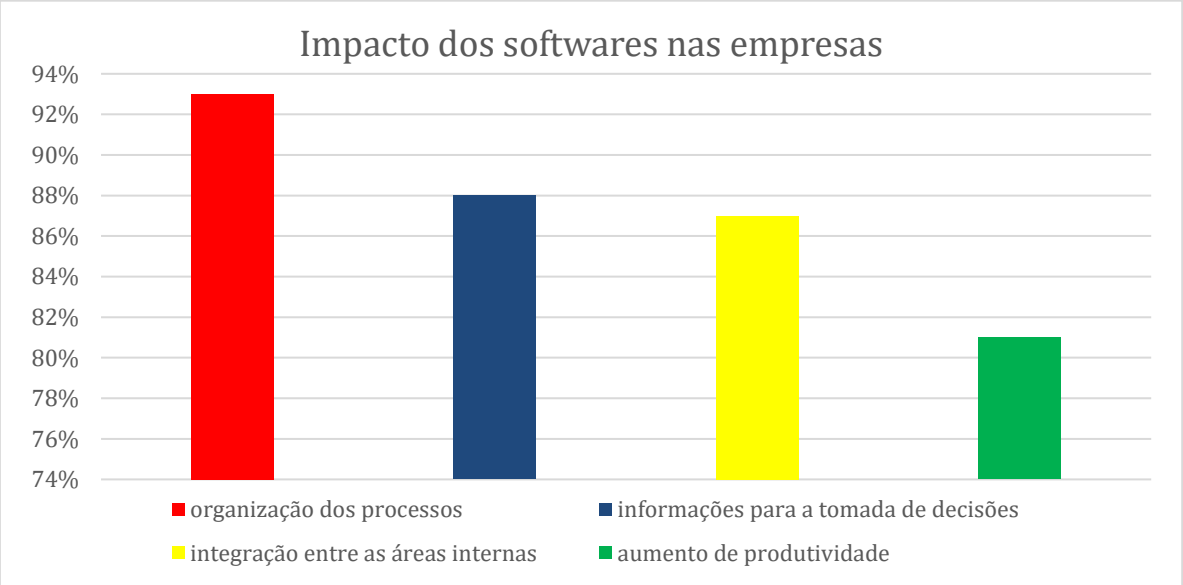
Porte Empresa	Software sob Encomenda
melhoria de processos e procedimentos internos	35%
ganho de produtividade e eficiência	22%
equações por exigência da lei	19%

Fonte: elaborado pelos autores.

Quanto aos impactos que os novos softwares introduzidos trouxeram para a empresa, 93% delas afirmaram que a iniciativa melhorou a organização dos processos realizados pela empresa, uma indicação de que os impactos mais evidentes das tecnologias estão localizados em seus processos internos.

Em segundo lugar está o impacto na produção de melhores informações para a tomada de decisões, que obteve 88% das menções das empresas. Outro impacto citado foi a maior integração entre as áreas internas da empresa na realização de suas atividades (87%), seguido pelo aumento de produtividade (81%) dados esses expostos no Gráfico 1-Impacto dos Softwares nas Empresas. Para 88% das empresas de grande porte, esse aumento da produtividade foi o principal impacto, 10 pontos percentuais acima do apresentando pelas empresas de pequeno porte como visto no Gráfico 1-Impacto dos Softwares nas Empresas.

Gráfico 1-Impacto dos Softwares nas Empresas



Fonte: elaborado pelos autores.

Através desta pesquisa de mercado a empresa 2vjmig obteve uma base do que precisa se obter e como será realizado o procedimento para dar início ao projeto Desktop, sendo objetivado usar um modelo incremental para atender melhor às necessidades do software feito sob encomenda pelo mercado Bom Preço.

5 A EMPRESA 2VJMIG

A empresa 2vjmig tem como objetivo buscar a excelência no desenvolvimento de seus Softwares para o mercado de trabalho, trazendo reconhecimento profissional e melhorias para a atual tecnologia existente, fazendo assim tornar-se uma grande potência no ramo da tecnologia.

Desenvolve softwares com máxima segurança e com total dedicação dos desenvolvedores, dando todo suporte e atenção necessário aos clientes.

Os seus sistemas são desenvolvidos seguindo todas as normas disponíveis, atendendo os requisitos pedidos por seus clientes para maior satisfação do mesmo fazendo assim gerar um produto de qualidade.

A política da empresa consiste em criar sistemas práticos, rápidos e seguros para seus clientes, trabalhando com dedicação e ética para atender as necessidades do mesmo, há um grande potencial ético por parte da empresa e dos desenvolvedores ao manter sigilo das informações referentes as empresas contratantes. Provendo suportes em relação aos seus produtos, mantendo assim o ciclo de vida de todos os seus softwares, a 2VJMIG consegue a satisfação de seus contratantes e a conceituação do nome da empresa no mercado.

6 ENGENHARIA DE SOFTWARE

O modelo incremental combina elementos do modelo cascata aplicado de maneira iterativa, aplica sequencias lineares de uma forma racional à medida que o tempo passa, sendo assim optou-se por esse modelo para ter um software moldado conforme o cliente deseja. É possível com esse modelo entregar parte do projeto, antes que o mesmo seja concluído totalmente, desta maneira nota-se que o projeto está de acordo com aquilo esperado pelo cliente obtendo uma vantagem maior perante a falha do projeto estabelecendo assim, ser um software de qualidade.

Um projeto onde a qualidade do software é essencial requer um bom planejamento, que auxilia a equipe de desenvolvedores a proceder em relação ao software, podendo ser prescrito em tabelas onde encontra-se os requisitos e funcionalidades do sistema, como exemplo a Tabela 4 - Variáveis do Cadastro de Funcionário, Tabela 5 - Variáveis do Cadastro de Cliente, Tabela 6 - Variáveis do Cadastro de Fornecedor e a Tabela 7 - Variáveis do Cadastro de Produtos.

Tabela 4 - Variáveis do Cadastro de Funcionário

Variável	Descrição	Tipo	RF	RNF
cvNomeFunc	Variável responsável por guardar o nome do funcionário	char	X	
cvCargo	Variável que armazena o cargo do funcionário	Struct	X	
cvEndereco	Variável responsável por guardar o endereço do funcionário	char	X	
cvCelular	Variável responsável por guardar um número de celular do funcionário	char	X	
			X	

Fonte: elaborado pelos autores.

Tabela 5 - Variáveis do Cadastro de Cliente

Variável	Descrição	Tipo	RF	RNF
cvCpf	Variável responsável por armazenar o CPF do cliente	char	X	
Cliente*cProximo	Váriável que indica o cadastro de um próximo cliente	Struct	X	
cvEndereco	Variável que guarda o endereço do cliente cadastrado	char	X	
cvTelefone	Variável que guarda um número de telefone	char	X	
cvData_nascimento	Variável que guarda a data de nascimento do cliente	char	X	

Fonte: elaborado pelos autores.

Tabela 6 - Variáveis do Cadastro de Fornecedor

Variável	Descrição	Tipo	RF	RNF
iCodigoProduto	Variável que guarda o código do produto	int	X	
cvCnpj	Variável que guarda o CNPJ do fornecedor	char	X	
Fornecedor*fProximo	Variável que indica o cadastro de um novo fornecedor	Struct	X	
cvNomeFantasia	Variável que guarda o nome da empresa do fornecedor	char	X	
cvEmail	Variável que guarda o e-mail do fornecedor	char	X	

Fonte: elaborado pelos autores.

Tabela 7 - Variáveis do Cadastro de Produtos

Variável	Descrição	Tipo	RF	RNF
iQtdEstoque	Variável que guarda a quantidade de produto em estoque	Int	X	
fValor	Variável responsável por guardar o valor do produto cadastrado	int	X	
iCodigoFornecedor	Variável responsável por guardar o código do fornecedor cadastrado	int	X	
Produto *pProximo	Variável que indica o cadastro de um novo produto	Struct	X	
cvCodigo	Variável que cadastra o código do produto	char	X	

Fonte: elaborado pelos autores.

Em uma conversação com o cliente foram levantados requisitos aos quais o software permita realizar cadastros, exclui-los e alterara-los dispondo a visualização dos dados cadastrados sendo eles os de funcionários, produtos, fornecedores e clientes.

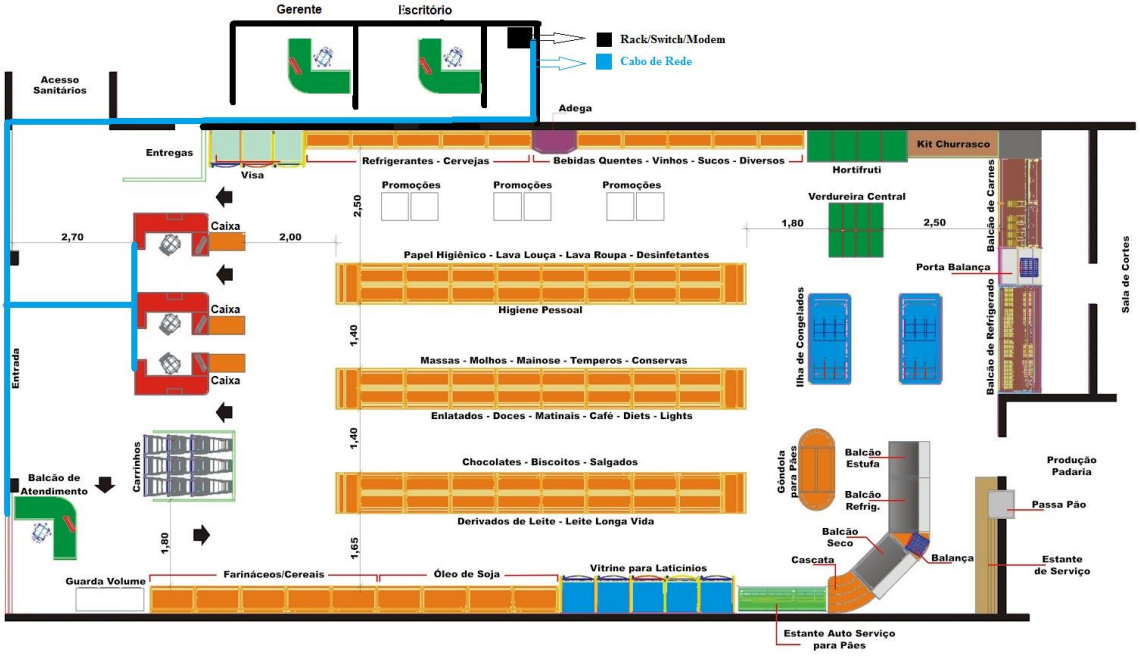
O software exibirá a quantidade de todos os produtos e conterá uma função de estoque de segurança para assim obter um controle do mesmo. Permitirá a venda desses produtos e caso aja problemas o usuário poderá realizar o cancelamento da mesma antes de finaliza-la, será disposto um controle das vendas feitas em que se poderá visualiza-las por data específica ou todas as vendas feitas pelo software.

Será disposto um login para o sistema afim de manter um controle para o gerente em que será destinado que o funcionário tenha acesso somente a função que lhe será disponibilizada.

7 REDES DE COMPUTADORES

A empresa 2VJMIG vendo necessário o uso de uma rede de computadores para que assim o sistema que será desenvolvido possa interagir entre os computadores, propõe ao Bom preço seus serviços quanto a implantação de rede onde irá interligar os computadores do caixa aos dos setores da Administração e da Gerencia objetivando usar uma topologia de rede em estrela, onde o Switch ficara dentro de um Rack que ficará na sala de administração como pode ser visto na Figura 1 - Esquema de Rede.

Figura 1 - Esquema de Rede



Fonte: elaborado pelos autores.

Os cabos da rede foram passados por baixo do chão em um eletroduto rígido, afim de evitar acidentes e rompimento dos mesmos, os equipamentos usados para a implantação da rede estão expostos na Tabela 8 - Lista e valor Dos Equipamentos da Rede, onde a implantação dessa rede totalizará em um valor de R$ 5.100,94, sendo 1.644,44 do equipamento e 3.456,50 da mão de obra da equipe.

Tabela 8 - Lista e valor Dos Equipamentos da Rede

Equipamento	Valor
Cabo de Rede Cat5e	300Mt, R$ 214,31
2 Switch	R$ 520,90 (cada)
Modem	R$ 188,90
Conector RJ45	50 Uni, R$ 18,99
Rj45 fêmea	15 Uni, R$ 119,49
Eletroduto rígido (5 barras de 3m)	Barra de 3m, R$ 16,19

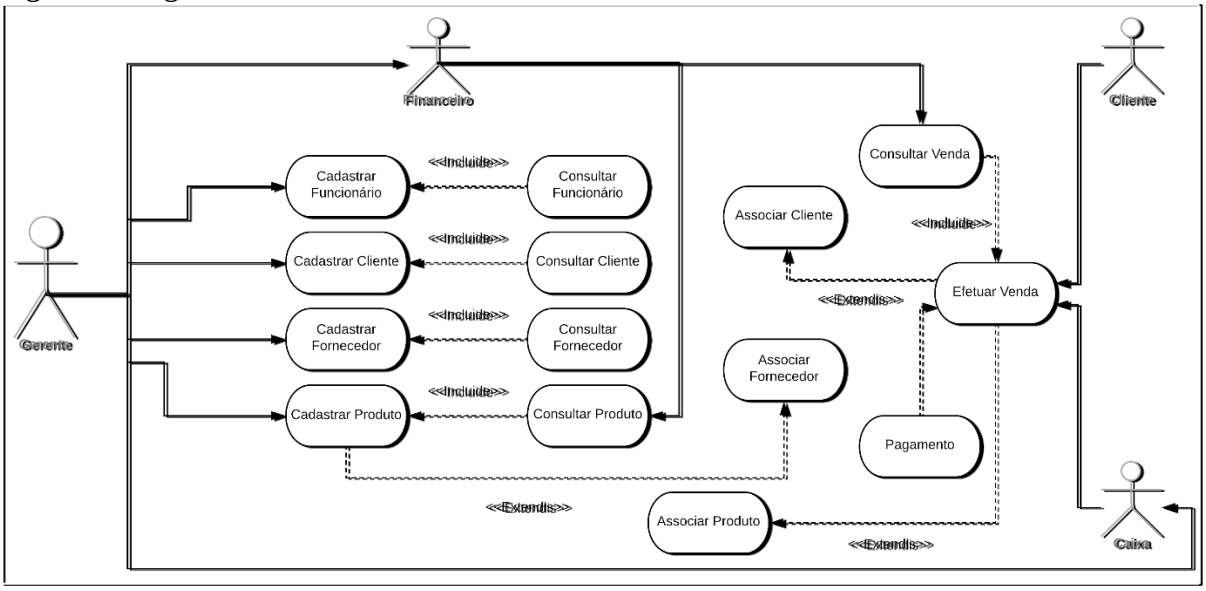
Fonte: elaborado pelos autores.

O mercado Bom Preço conta com uma configuração DHCP, para que em uma eventual queda de energia ou perca de configuração por outro motivo os computadores se reconectem com seus Ip's automáticos com máscara de rede: 255.255.255.0.

8 SOFTWARE iMerc

Em uma reunião com o cliente, a empresa 2vjmig obteve a proposta de desenvolver um software que agilizasse o fluxo do caixa para os clientes, fazendo assim evitar que o mercado necessite contratar novos funcionários e que não haja clientes insatisfeitos com a demora ao pagar sua compra. A empresa destacou as funcionalidades e requisitos que o software deveria obter, sendo descritos em um diagrama de caso como visto na Figura 2-Diagrama Caso de Uso em que descreve como funcionara o programa, auxiliando assim os desenvolvedores.

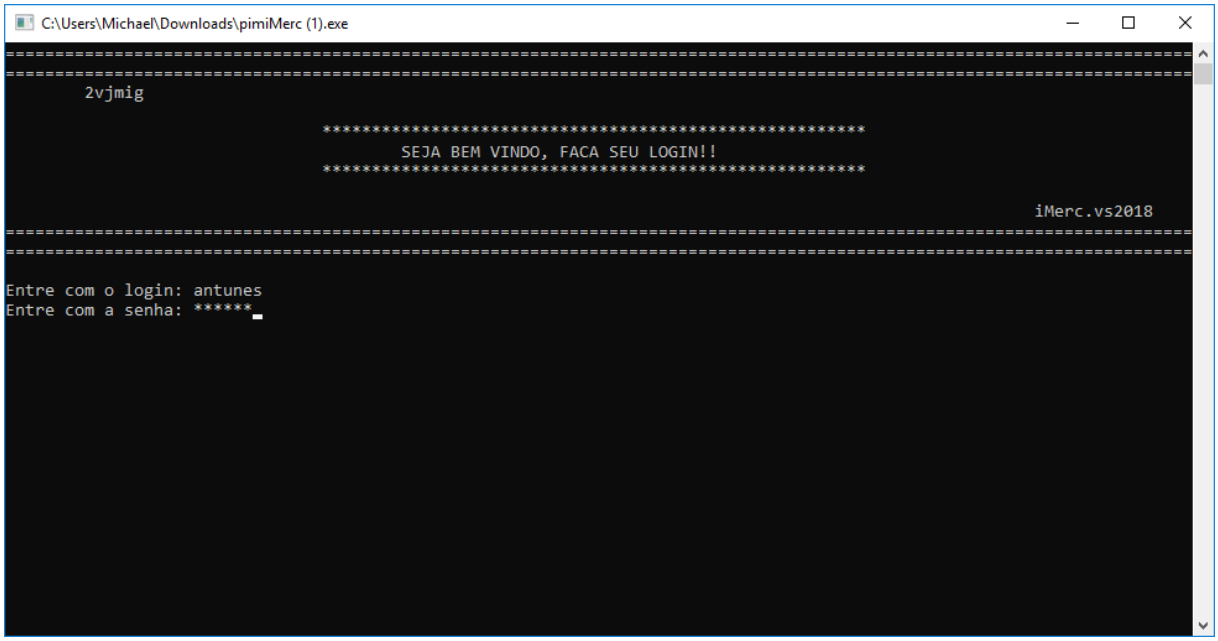
Figura 2-Diagrama Caso de Uso



Fonte: elaborado pelos autores

O software, que foi nomeado de iMerc pelos desenvolvedores, tem como princípio a hierarquia de usuários, subdivididos para que cada usuário faça algo em relação ao software, barrados por uma tela de login e senha como visto na Figura 3 - Login e senha que destina esse usuário à função a ser usada do programa.

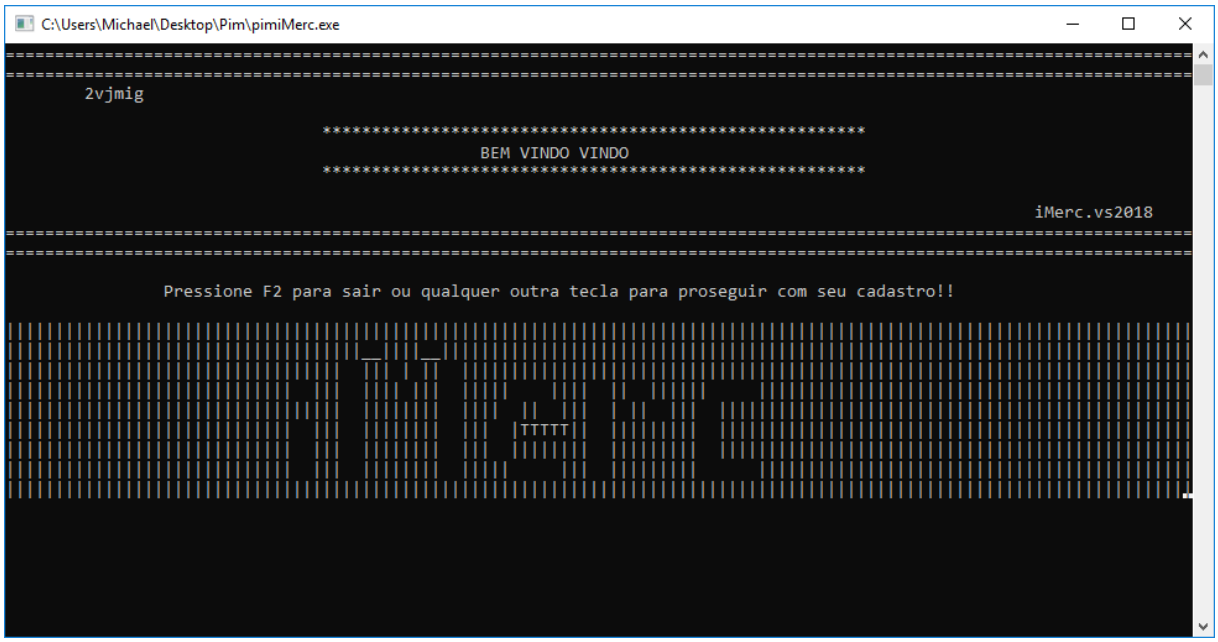
Figura 3 - Login e senha



Fonte: elaborado pelos autores.

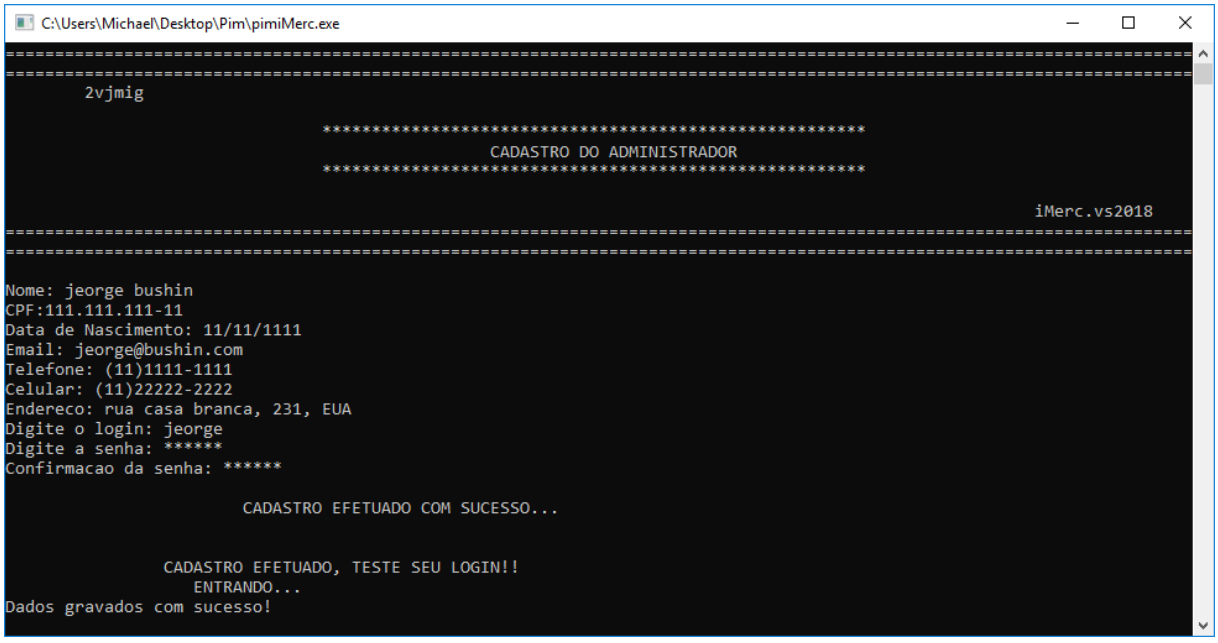
Caso seja a primeira vez de uso do iMerc, ele encaminhará o usuário, que no caso será destinado como Administrador, a fazer seu cadastro como visto na Figura 4 - Primeiro acesso e Figura 5 - Primeiro Cadastro, salvando esses dados em um arquivo para que na próxima utilização ele não necessite refazê-los.

Figura 4 - Primeiro acesso



Fonte: elaborado pelos autores.

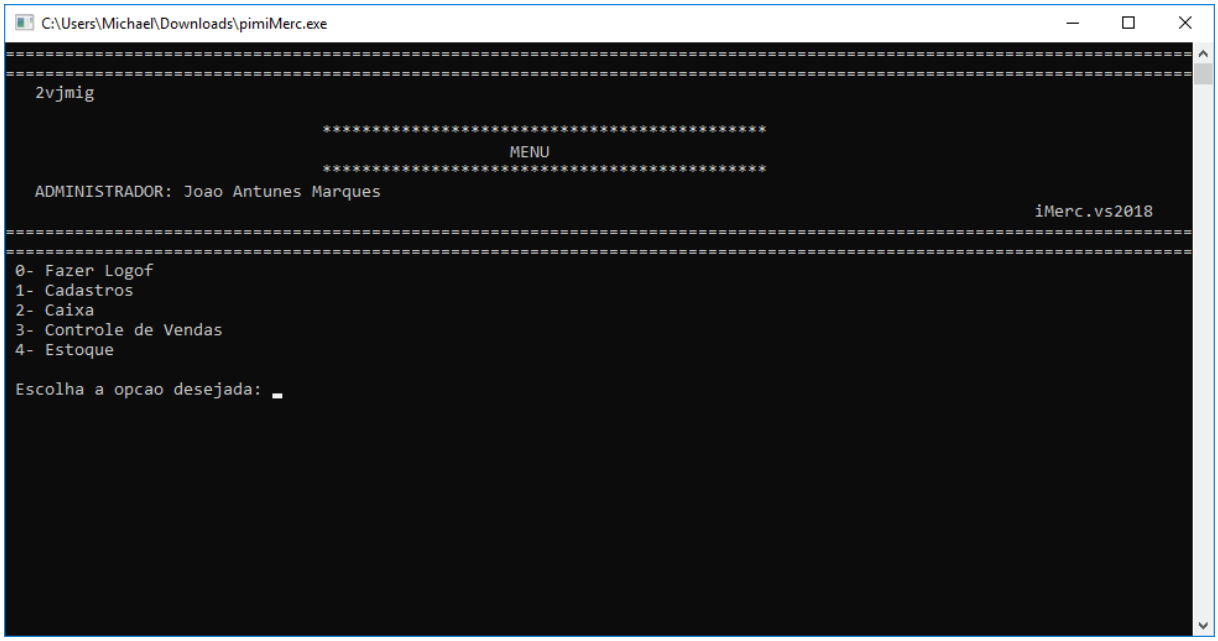
Figura 5 - Primeiro Cadastro



Fonte: elaborado pelos autores.

O login do usuário Administrador é aquele que tem acesso total ao programa podendo ser descrito como o proprietário do estabelecimento, efetuando os demais cadastros sendo eles o cadastro de funcionário (caixa), o cadastro de fornecedores, o cadastro de produtos e o cadastro de clientes, ele tem acesso ao caixa do programa podendo assim efetuar as vendas e tem acesso as listas de todos os cadastros incluindo a de vendas finalizadas sendo possível ter o controle de vendas e de estoque como visto na Figura 6 - Menu Administrador.

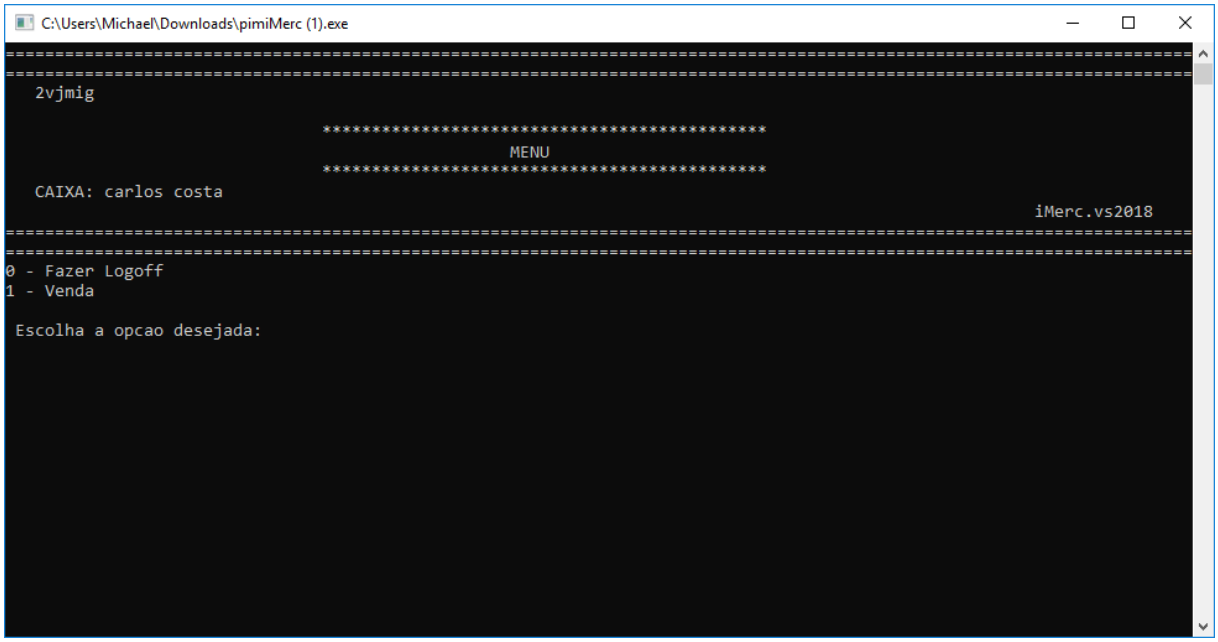
Figura 6 - Menu Administrador



Fonte: elaborado pelos autores.

O login do usuário Caixa é aquele que terá acesso somente ao caixa, sendo lhe permitido somente a parte de venda dos produtos como visto na Figura 7 - Menu caixa.

Figura 7 - Menu caixa

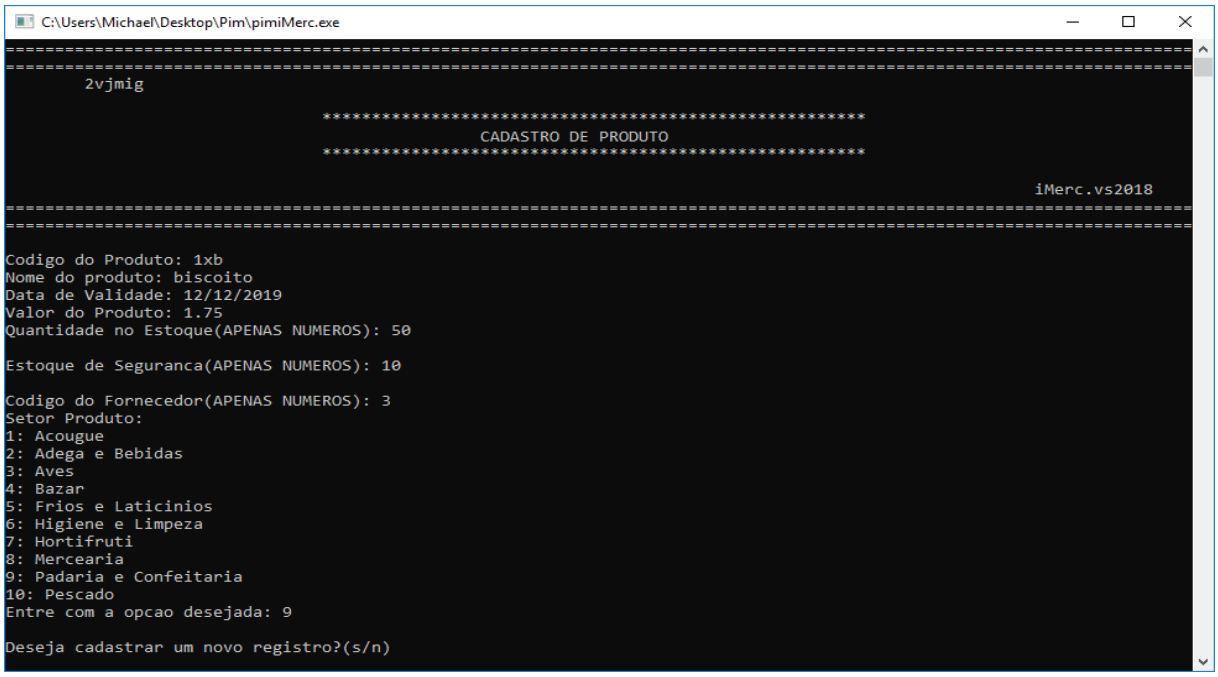


Fonte: elaborado pelos autores.

Os usuários implicam na segurança do sistema, fazendo com que o mesmo se torne mais seguro para o proprietário fazendo assim ter um melhor gerenciamento.

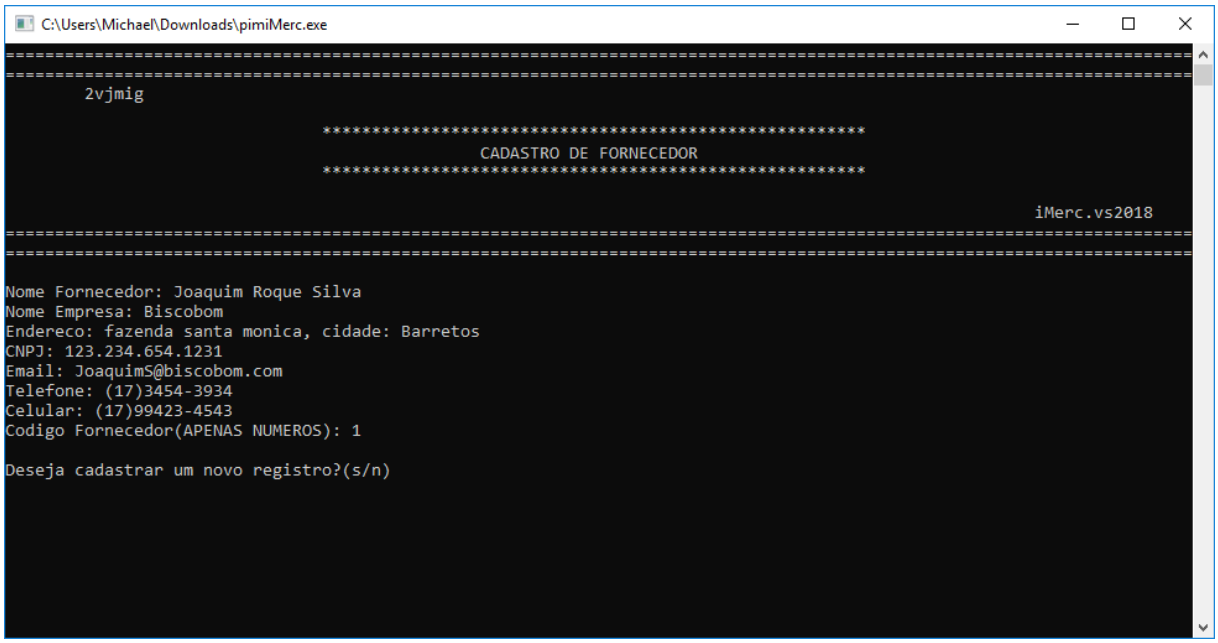
Para que seja possível a alimentação do estoque assim como em várias sistema do segmento de varejo é necessário o cadastro dos produtos (Figura 8 - Cadastro de Produto) e dos fornecedores (Figura 9 - Cadastro de Fornecedor) desses produtos, para que assim seja possível repor o estoque agilmente ao ter o cadastro do fornecedor, essas funções de cadastro são destinadas ao usuário Administrador.

Figura 8 - Cadastro de Produto



Fonte: elaborado pelos autores.

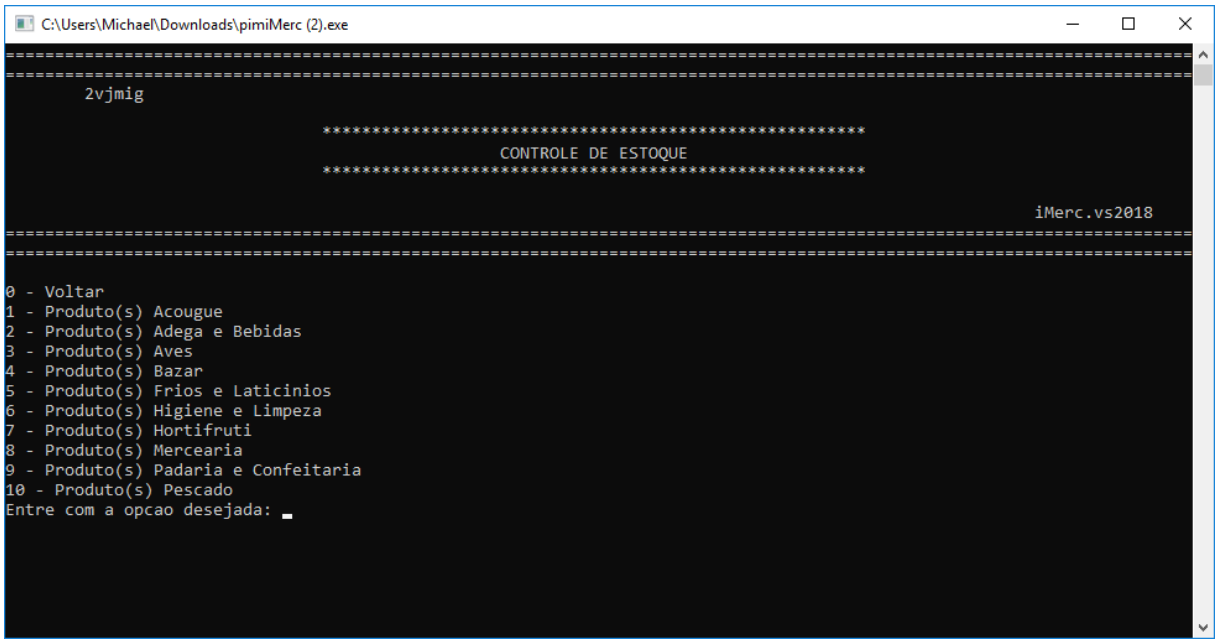
Figura 9 - Cadastro de Fornecedor



Fonte: elaborado pelos autores.

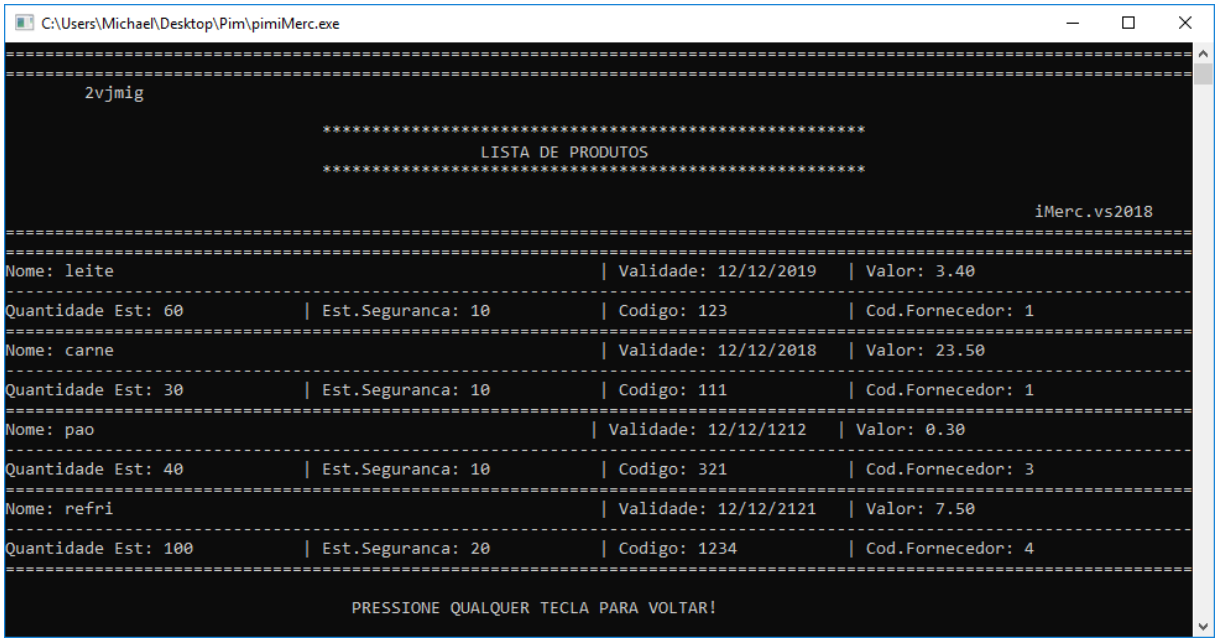
O controle de estoque (Figura 10 – Controle de Estoque) criado para verificar a quantidade de produtos e o estoque de segurança do mesmo (Figura 11 - Lista de Produtos), dará uma vantagem ao permitir que não falte nada no mercado, ou seja, o sistema emitirá uma mensagem alertando que o estoque de segurança foi atingido.

Figura 10 – Controle de Estoque



Fonte: elaborado pelos autores.

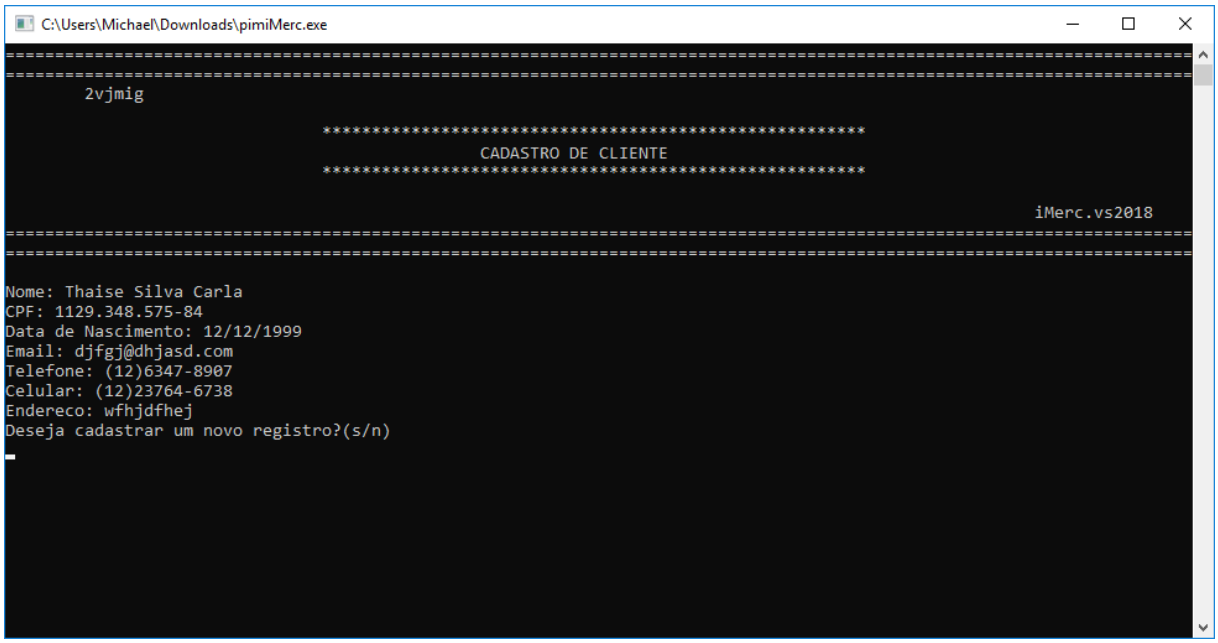
Figura 11 - Lista de Produtos



Fonte: elaborado pelos autores.

A parte de cadastro de clientes (Figura 12 - Cadastro de Cliente), podendo parecer ser pouco viável para o programa, torna possível aumentar a frequência do mesmo de maneira estratégica em que com o seu cadastro feito ele possa ganhar desconto em suas compras, isso de acordo com o valor gasto em mercadorias, essa função fica a cargo do administrador e é um requisito não funcional podendo ou não ser aderido por ele.

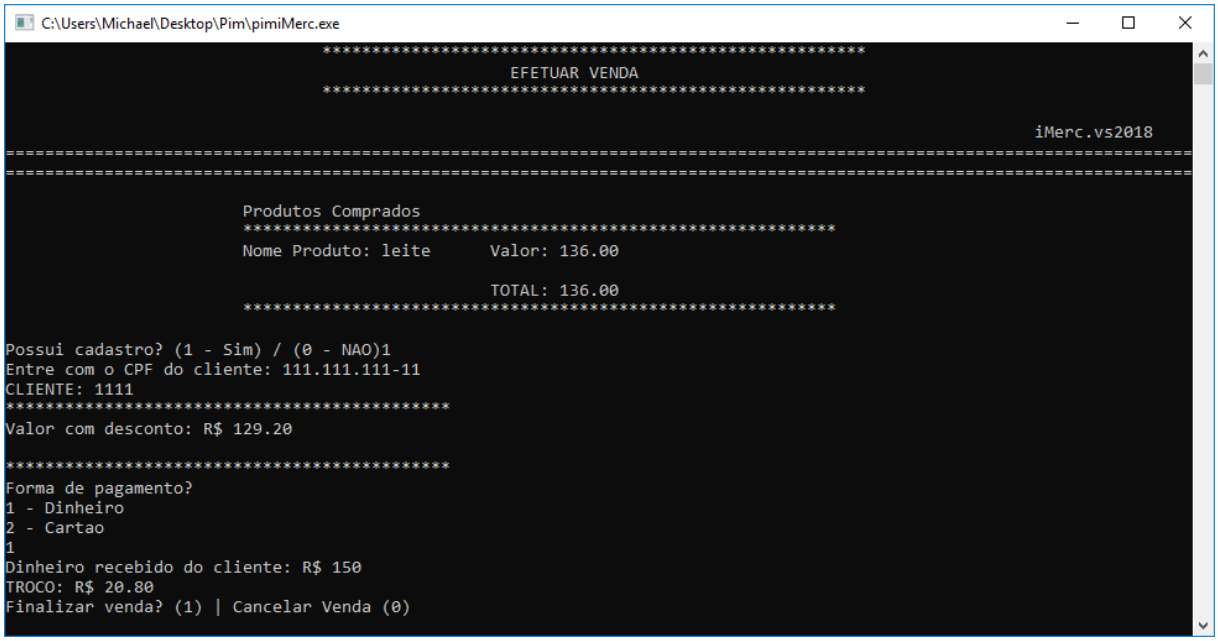
Figura 12 - Cadastro de Cliente



Fonte: elaborado pelos autores.

A parte de venda, normalmente utilizada pelo usuário caixa, permite efetuar a compra do cliente listando cada item e a quantidade deles baseando-se em uma nota fiscal onde contém o que foi comprado e o valor total da compra. Ao finalizar a compra há duas possíveis formas de pagamento sendo elas em dinheiro ou cartão. A forma de pagamento com o cartão é feita através de outro software, ou seja, a venda é finalizada e registrada apenas, mas nos planos futuros será possível através do iMerc associar essa forma. O pagamento em dinheiro contará com a funcionalidade de calcular o troco do cliente auxiliando assim o funcionário, logo será finalizada a compra, ao finalizar uma compra o programa a registrará para que seja possível obter um controle dessas vendas como visto na Figura 13 - Venda .

Figura 13 - Venda concluída



Fontes: elaborado pelos autores.

O controle de vendas foi desenvolvido de modo a dar uma dinâmica ao Administrador em que ele possa ver todas as vendas realizadas pelo programa, pesquisar vendas especificas separadas por somente dia, mês ou ano para que assim ele possa analisar esses dados e fazer uma tomada de decisão para o mercado ou mesmo verificar algum erro iminente feito pelo mal uso do software.

9 IMPLEMENTAÇÃO

A implementação do projeto foi feita em linguagem C pois é uma linguagem que permite manipular dados simples, obter acesso direto à memória e trabalhar com funções matemáticas, ficheiros entre outras, sendo necessário para tal a inclusão das bibliotecas ao qual no projeto utilizou-se as: stdio.h, stdlib.h, string.h e time.h.

O sistema contém várias funções para obter uma estruturação mais organizada do código fonte, onde caso apareça algum erro no sistema fique mais fácil de acha-lo e corrigi-lo.

Foi implementado ao sistema campos obrigatórios exemplo CPF, datas, nomes entre outros campos além disso contem textos de apoio, para assim o usuário inserir os dados corretos evitando o uso inadequado do software.

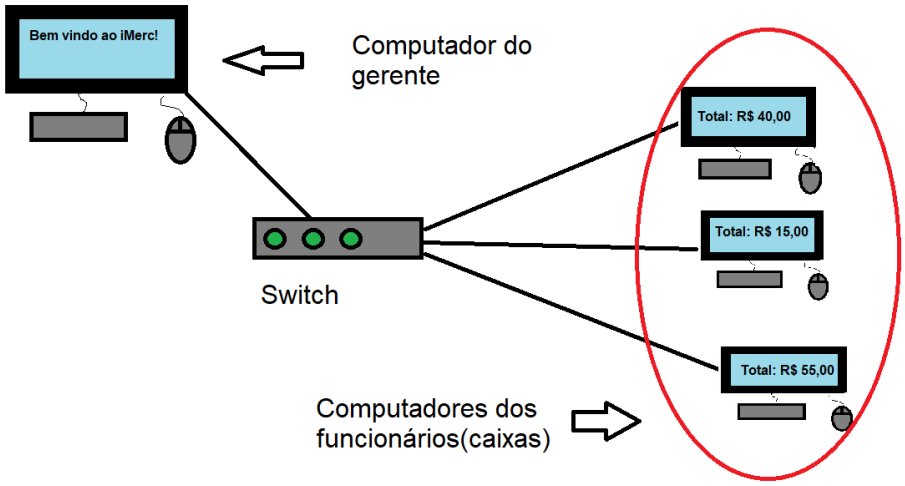
Foram gastas 90 horas para finalizar o projeto utilizando uma equipe de seis pessoas, onde três eram programadores, dois da parte de teste e um coordenador de projeto, onde mantinha um controle do tempo gasto no sistema para assim entregar um produto de qualidade dentro do prazo determinado entre a empresa 2VJMIG e o mercado Bom Preço.

10 IMPLANTAÇÃO

O software foi projetado com base em um Sistema Operacional Windows 7 ou acima, visto que o mercado utiliza S.O's da Microsoft, e necessita de requisitos mínimos de forma em que o desempenho do software não seja afetado. Para uma boa performance o computador ao qual o iMerc será instalado e executado, deverá conter as seguintes exigências mínimas sendo que 2GB de memória RAM; 10 GB de espaço livre em disco e Processador Dual Core sejam necessários.

O iMerc deve ser instalado em um computador que possua conexão com à rede local do estabelecimento, para que assim haja a troca de informações entre os computadores dos empregados com o sistema principal do gerente exemplo a Figura 14 - Diagrama de implantação.

Figura 14 - Diagrama de implantação



Fonte: elaborado pelos autores.

A implantação do iMerc foi feita pela empresa 2VJMIG onde após instalar o sistema testou o mesmo de modo a evitar possíveis problemas causados pela má implantação. A empresa 2VJMIG apresentou ao cliente os custos visto na Tabela 9 - Etapas da implantação, afim de mostra o empenho e dedicação de sua empresa para chegar a um acordo com o cliente em que além de manter suporte técnico possa vir a implementar mais funcionalidades no sistema caso o mesmo queira algo novo.

Tabela 9 - Etapas da implantação

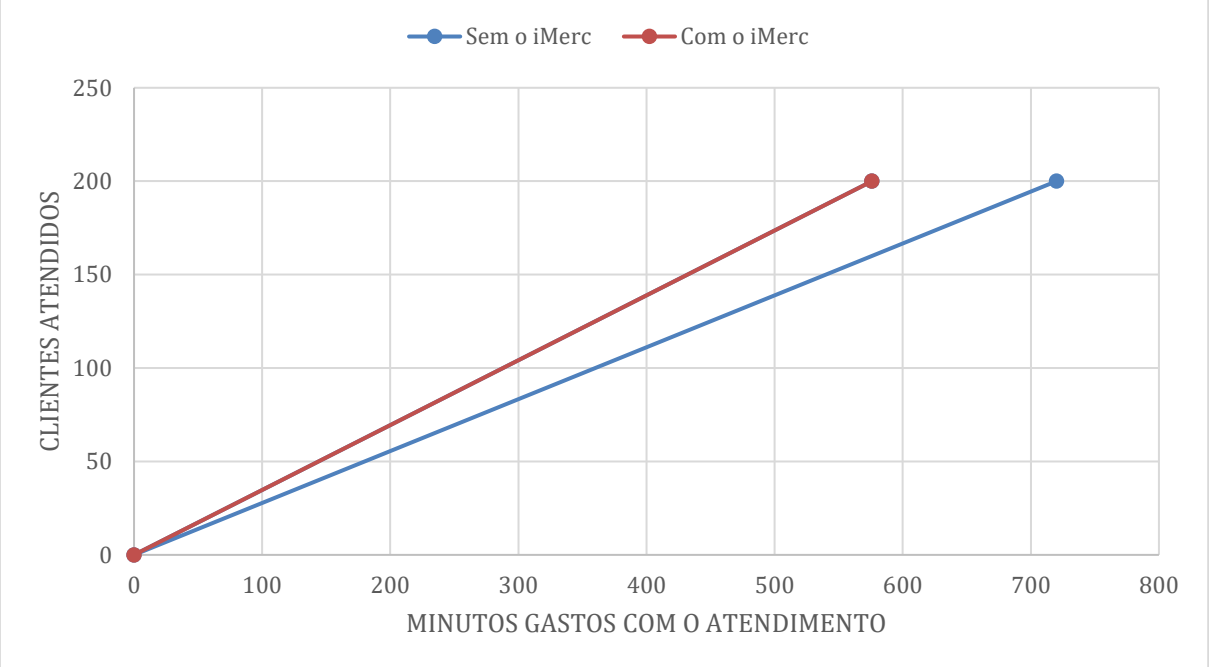
Etapa	Atividade	Horas implantação
1	Planejamento da implantação	2
2	Projeto piloto	8
3	Desenvolvimento	90
3.1	Diálogo com o cliente	7
3.2	Engenharia	9
3.3	PCP	19
3.4	Equipe	10
3.5	Custos	21
3.6	Documentação	14
3.7	Qualidade	10
	Total de horas	100
	Preço/hora(media)	R$ 95
	Investimento total	9.500
	Prazo de implantação estimado em semanas	13

Fonte: elaborado pelos autores.

11 RESULTADOS

A 2VJMIG após implantar o iMerc, realizou uma análise do fluxo de caixa para comparar com os dados anteriores ao sistema em que verifica a média de clientes por dia e o tempo gasto para atender esses clientes como visto no Gráfico 2 - Tempo de Atendimento. Notou-se nessa analise realizada que o tempo de atendimento em relação à média de clientes, diminuiu em aproximadamente 20% fazendo assim o software corresponder às expectativas do Bom Preço em relação ao fluxo de caixa.

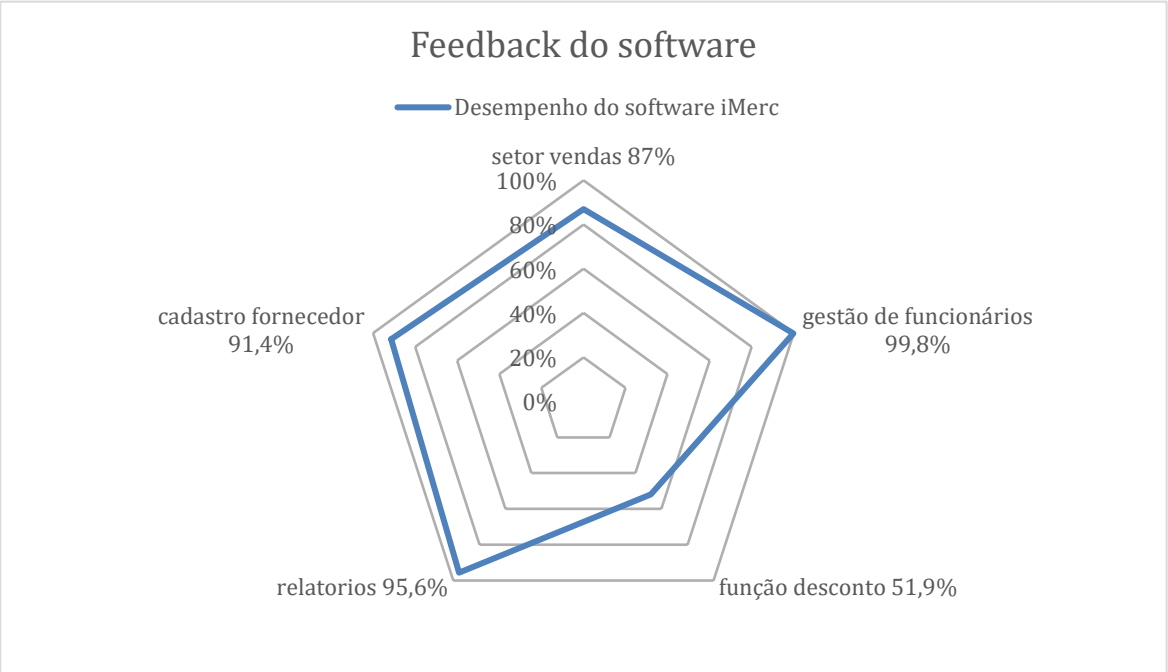
Gráfico 2 - Tempo de Atendimento



Fonte: elaborado pelos autores.

Em uma reunião marcada para expor essa informação, o cliente mostra satisfação quanto ao produto adquirido, onde ele retorna um feedback citando que as funções do sistema otimizaram as atividades do mercado, mencionando que o setor de vendas teve uma eficiência de 87%, o gerenciamento de funcionários facilitou 99,8% o setor administrativo, os relatórios agilizaram em 95,6% o setor de finanças, o cadastro de fornecedor facilitou em 91,4% a comunicação e a negociação com o mesmo, com a liberação de descontos foi possível atrair mais clientes, notando que em determinado período de tempo houve um aumento de 51,9%, dados esses em relação ao iMerc visto no Gráfico 3- Feedback do software.

Gráfico 3- Feedback do software



Fonte: elaborado pelos autores.

Analisando o feedback feito pelo cliente, notou-se que os resultados obtidos eram os esperados, fazendo assim com que o iMerc tenha uma ótima adesão, podendo ser lançadas novas atualizações com novas funcionalidades conforme sendo solicitados pelo cliente ou pacotes de atualizações, funcionado como fonte de renda para a empresa 2vjmig e mantendo o ciclo de vida do produto.

12 CONCLUSÃO

Conclui-se que o objetivo do presente trabalho foi atingido ao notar que a Engenharia de Software foi apresentada como princípio para o desenvolvimento do iMerc, onde foi visto o modelo organizacional, os requisitos do sistema e uma tabela para auxiliar os programadores com o desenvolvimento do mesmo, descritos no tópico 6. Logo a Lógica de Programação foi tratada no desenvolvimento do sistema no tópico 8, cujo um dos objetivos era agilizar o fluxo de caixa do mercado Bom Preço, onde no tópico 12 foi visto que os conhecimentos obtidos em Matemática constataram que esse objetivo foi atingido tendo um percentual de queda no tempo de atendimento em aproximadamente 20%.

O sistema, como visto no tópico 11, necessitaria de uma rede para a troca de informações, onde no tópico 7 está exposto o aprendizado de Redes de Computadores que indica uma possível rede disponibilizada pela 2VJMIG. A Ética foi tratada no tópico 5 onde mostra as políticas da empresa ao tratar seus clientes não expondo informações dos mesmos.

A Acessibilidade e a Sustentabilidade foram citadas afim de promover que a tecnologia pode contribuir com estes aspectos além de servir como estratégia para possuir mais clientes contribuindo tanto com o mercado quanto com os clientes e a natureza.

12.1 Planos futuros

De acordo com algumas recomendações do cliente e pendencias do iMerc, será disposto algumas melhorias no sistema facilitando e aperfeiçoando a experiencia do usuário tornando o software o mais prático e usual possível.

Será disposto um pacote com a funcionalidade de pagamentos em cartões, tanto bancários como vales dispondo a transação como debito ou credito. Melhoria na interface tornando o sistema mais compreensível dispondo o layout mais explicativo, mascaras e travamento de todos os campos para assim evitar a inserção de dados incorretos.

REFERÊNCIAS

CORACCINI, Raphael; **6 Tecnologias aplicadas ao varejo que se destacaram em 2018,** disponível em: <https://portalnovarejo.com.br/2018/05/6-tecnologias-aplicadas-apas-2018/>. Acesso em 15 de setembro de 2018.

JUNQUEIRA, Gabriel; **6 principais motivos de implantar um software de gestão no mercado,** disponível em: <https://www.infovarejo.com.br/motivos-software-de-gestao-no-supermercado/.>. Acesso em 19 de setembro de 2018.

LOURENÇO COUTO, João; **A importância dos supermercados para a economia comercial,** disponível em: <https://www.webartigos.com/artigos/a-importancia-dos-supermercados-para-a-economia-comercial/85336/>. Acesso em 22 de setembro de 2018.

LOURENÇO COUTO, João; **A importância dos supermercados para a economia comercial,** disponível em: <http://artigos.netsaber.com.br/resumo_artigo_70939/artigo_sobre_a-importancia-dos-supermercados-para-a-economia-comercial>. Acesso em 01 de outubro de 2018.

MARTINI, Anselmo; **A importância da tecnologia aplicada no varejo,** disponível em: <https://www.administradores.com.br/noticias/negocios/a-importancia-do-uso-da-tecnologia-aplicada-no-varejo/122229/>. Acesso em 11 de setembro de 2018.

REDAÇÃO; Os 10 principais motivos que levam uma empresa a falência, disponível em: <http://www.administradores.com.br/noticias/negocios/10-principais-motivos-que-levam-uma-empresa-a-falencia/118239/>. Acesso em 15 de setembro de 2018.

REDAÇÃO; Supermercados poderão ser obrigados a ter carrinhos adaptados para crianças deficientes, disponível em: <https://www.al.sp.gov.br/noticia/?id=388204>. Acesso em 15 de setembro de 2018.

SACONI, Rose; **Como era São Paulo sem supermercado,** disponível em: <https://acervo.estadao.com.br/noticias/acervo,como-era-sao-paulo-sem-supermercado,9180,0.htm>. Acesso em 25 de agosto de 2018.

S/A; Sustentabilidade; disponível em: <https://escolakids.uol.com.br/sustentabilidade.htm>. Acesso em 30 de agosto de 2018.

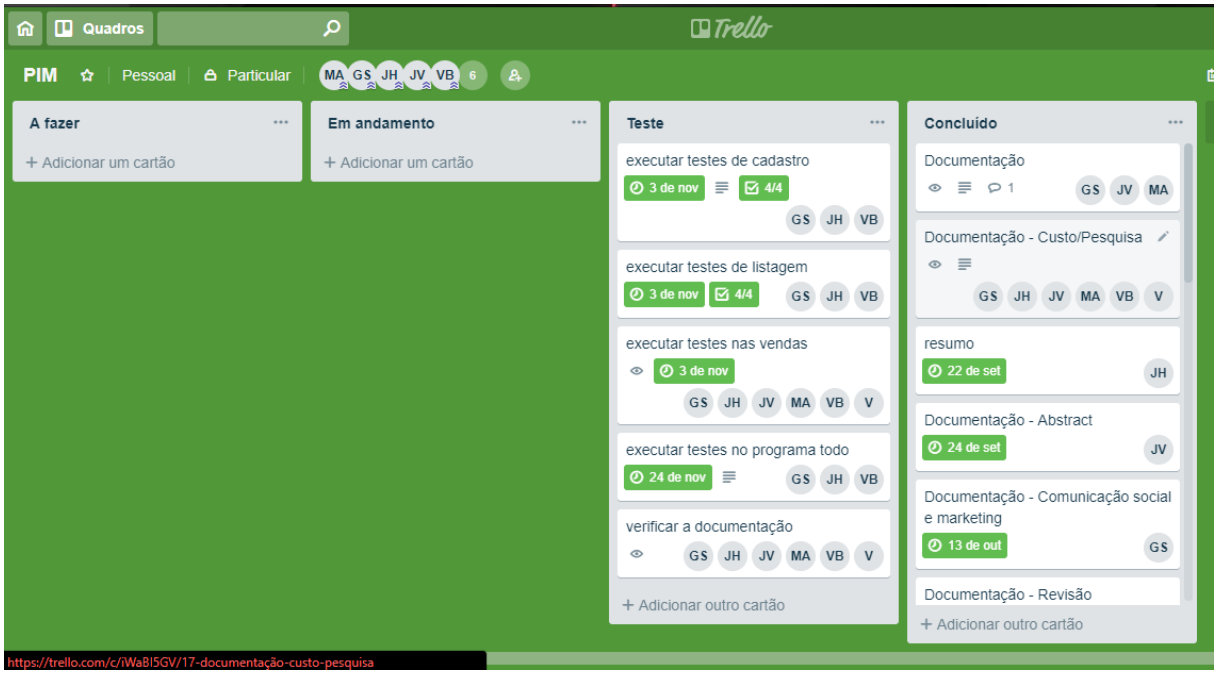
S/A; Projetos sustentáveis: por que a sustentabilidade é importante para o seu negocio?, disponível em: <https://descola.org/drops/projetos-sustentaveis-por-que-a-sustentabilidade-e-importante-para-o-seu-negocio/>. Acesso em 28 de agosto de 2018.

SEBRAE; Pesquisa de Mercado: Motivos e Impactos do Uso de Softwares, disponível em: <http://www.sebraemercados.com.br/pesquisa-de-mercado-motivos-e-impactos-do-uso-de-softwares/>. Acesso em 01 de outubro de 2018.

APÊNDICE

Visto que todo projeto para possuir um bom planejamento utiliza-se de técnicas para se organizar. Para desenvolver o software iMerc, referido no presente trabalho, foi utilizado um software de organização e comunicação chamado Trello, onde como visto na Figura 15 - TRELLO gerencia o projeto por tarefas sendo elas subdivididas em a fazer, em andamento, teste e tarefa concluída, podendo atribuir mais cartões caso necessário.

Figura 15 - TRELLO



Fonte: elaborado pelos autores.

O Trello possui a função de associar membros a tarefas como mostra a Figura 16 - Tarefa, fazendo assim obter uma divisão de tarefas como forma de agilizar as mesmas afim de obter êxito e agilidade na hora de obter o produto final.

Figura 16 - Tarefa

Fonte: elaborado pelos autores.

O Trello permite poder alterar ou acrescentar a qualquer hora os membros ou as tarefas sendo bem usual para projetos que sofrem constante mudanças, onde foi bem aproveitado para o desenvolvimento do iMerc que utilizando um modelo incremental estava suscetível a alterações.

O iMerc foi estruturado utilizando-se de funções para permitir um melhor manejo e elaboração do código fonte, facilitando assim a verificação e alteração caso necessário como visto na Tabela 10 - Codigo fonte a estruturação do mesmo.

Tabela 10 - Codigo fonte

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
typedef struct produto
{
    char cvNome[500];
    char cvValidade[100];
    float fValor;
    int iQtdEstoque;
    int iEstoqSeguranca;
    char cvCodigo[50];
    char cvSetor[200];
    int iCodigoFornecedor;
```

```
    struct Produto *pProximo;
} Produto;

typedef struct fornecedor
{
    char cvNome[101];
    char cvNomeFantasia[101];
    char cvEndereco[101];
    char cvCnpj[20];
    char cvEmail[101];
    char cvTelefone[101];
    char cvCelular[101];
    int iCodigoProduto;
    struct Fornecedor *fProximo;
} Fornecedor;

typedef struct cliente
{
    char cvNomeCli[200];
    char cvCpf[15];
    char cvData_nascimento[101];
    char cvEmail[101];
    char cvTelefone[101];
    char cvCelular[101];
    char cvEndereco[101];
    struct Cliente *cProximo;
} Cliente;

typedef struct funcionario
{
    char cvNomeFunc[101];
    char cvCargo[101];
    char cvCpf[15];
    char cvData_nascimento[101];
    char cvEmail[101];
    char cvTelefone[101];
    char cvCelular[101];
    char cvEndereco[101];
    char cvLogin[20];
    char cvSenha[07];
    struct Funcionario *funProximo;
} Funcionario;

typedef struct venda
{
      int iDia;
      int iMes;
      int iAno;
      int iCodVenda;
      float dTotalCompra;
    struct Venda *vendaProximo;
} Venda;

typedef struct itemVenda{
      char cvNomeItem[201];
      float fValorItem;
      struct ItemVenda *iItemProximo;
}ItemVenda;
```

```c
/* DECLARACAO DE STRUCTS */
Produto *pProdutoInicial = NULL;
Fornecedor *fFornecedorInicial = NULL;
Cliente *cClienteInicial = NULL;
Funcionario *fFuncionarioInicial = NULL;
Venda *vVendaInicial = NULL;
ItemVenda *iItemVendaInicial = NULL;
char cvAdm [10];
int iHaveLogin = 0;
char cvNFuncionario[200];
char cvCargoFunc[200];
char cvNomeCli[200];

/* DECLARACAO DE  FUNCOES */
void sair();
void defaultMessage();
void removidoSucesso(char cvNomeFuncao[], char cvNomeItem[]);
void editMessage(char cvFunction[]);
void alteradoSucesso(char cvContext[]);
void nullList();
void notFound(char cvName[]);
void nullDelete();
void listarProdutos();
void removeProduto();
void listarFornecedor();
void alteraFornecedor();
void removeFornecedor();
void listarCliente();
void alteraCliente();
void removeCliente();
void listarFuncionario();
void alteraFuncionario();
void removeFuncionario();
void CadastroLogin(char cvCadastroLogin[],char cvCadastroSenha[]);
void VerificarLogin();
void tela(char tela[]);
void listarVenda();
void listaItemVenda(float fTotalVenda);
void listarEstoque(char nome[]);

void verificaEstoqueSegurancao();

/* DECLARACAO DE FUNCOES QUE RETORNAM VALOR */
int menuPrincipal();
int menu();
int menuCaixa();
int menuListar();
int menuAlterar();
int menuRemover();
void menuVenda();
int menuEstoque();
char escolhaRegistro();
int menuCadastro();
int cadastrarProduto();
int alteraCodigoProduto();
int alteraNomeProduto();
int alteraValorProduto();
int alteraQtdProduto();
int alteraEstoqueSeg();
```

```
void listarEstoque();
int cadastrarFornecedor();
int cadastrarCliente();
int cadastrarFuncionario();
int venda();
float addItemVenda();
int addVenda(float totalVenda);
void addStructVenda(Venda *vNovaVenda);
void tiraEstoque();
void vendaRealizada(char mensagem[]);
int pesqVendaData();
int verificaCliente();

/* CHAR FUNCTIONS */
float pegaValorProd();
int FormatarData();
int FormatarTelefone();
int FormatarCelular();
int FormatarCPF();
int Nome();

/* Arquivos Functions */
int gravarProduto();
int gravarFuncionario();
int gravarLogin();
int gravarCliente();
int gravarFornecedor();
int gravarItemVenda();
int gravarVenda();
int gravarAdm(char adm[]);

int lerProduto();
int lerFuncionario();
int lerLogin();
int lerCliente();
int lerFornecedor();
int lerItemVenda();
int lerVenda();
int lerAdm(char adm[]);

int main(){
       //ler dados dos arquivos
       lerAdm(cvAdm);
       lerProduto();
       lerFuncionario();
       lerFornecedor();
       lerCliente();
       //lerItemVenda();
       lerVenda();
       VerificarLogin();

    return 0;
}

void sair(){
       gravarProduto();
    printf("\nObrigado por utilizar o programa!!!\n");
}
```

```
void defaultMessage(){
    printf("\nPor favor, entre com uma opcao valida!\n");
}

void editMessage(char cvFunction[]){
    printf("\nEntre com o nome do %s a ser alterado: ", cvFunction);
}

void alteradoSucesso(char cvContext[]){
       printf("\n\t\t\t%s foi alterado(a) com sucesso!!!\n", cvContext);
}

void removidoSucesso(char cvNomeFuncao[], char cvNomeItem[]){
       printf("\n\t\t\tO %s %s foi removido com sucesso!!\n", cvNomeFuncao,
cvNomeItem);
}

void nullList(){
    printf("\nNao ha registros para serem listados!!!\n");
    printf("\n\t\t\t\t   PRESSIONE QUALQUER TECLA PARA VOLTAR!");

}
void notFound(char cvName[]){
       printf("\n%s nao encontrado(a)!", cvName);
}
void nullDelete(){
    printf("\nNao ha registros para serem removidos!!\n");
}

char escolhaRegistro(){
    char cEscolha = 's';
    do{
       printf("\nDeseja cadastrar um novo registro?(s/n) \n");
       scanf(" %c", &cEscolha);
       if(cEscolha == 's' || cEscolha == 'S' || cEscolha == 'n' || cEscolha ==
'N') break;
       }while(cEscolha != 's' || cEscolha == 'S' || cEscolha != 'n' || cEscolha
!= 'N');

    return cEscolha;
}

void verificaEstoqueSegurancao(){
       Produto *pAux = pProdutoInicial;
       while(pAux != NULL){
              if(pAux->iQtdEstoque < pAux->iEstoqSeguranca){

printf("\n\n================================================================================
================================================================================
================================================================================
============\n");
                     printf("%s precisa que seu estoque seja reabastecido!!!\n",
pAux->cvNome);

printf("================================================================================
================================================================================
================================================================================
========\n\n");
             }
```

```
            pAux = pAux->pProximo;
        }
}

int menuPrincipal()
{

    int iOp;
     Funcionario *fAux = fFuncionarioInicial;
    system("cls || clear");

printf("================================================================================
================================================================================
================================================================================
========");
                                printf("\t   2vjmig\n\n");

       printf("\t\t\t\t*****************************************\n");
                           printf("\t\t\t\t\t\t   MENU\n");

       printf("\t\t\t\t*****************************************\n");
                       printf("   %s: ", cvCargoFunc);
                       printf("%s ", cvNFuncionario);
                           printf("\n\t\t\t\t\t\t\t\t\t\t\t\t\t\tiMerc.vs2018\n");

       printf("================================================================================
================================================================================
================================================================================
===============\n");
       verificaEstoqueSegurancao();
    printf(" 0- Fazer Logof\n");
    printf(" 1- Cadastros\n");
    printf(" 2- Caixa\n");
    printf(" 3- Controle de Vendas\n");
    printf(" 4- Estoque\n");
    printf("\n Escolha a opcao desejada: ");
    scanf("%d", &iOp);
    return iOp;
}

int menu()
{
    int iOp;
    do
    {
        iOp = menuPrincipal();
        switch (iOp)
        {
        case 0:
             system("cls || clear");
            VerificarLogin();
            break;
        case 1:
            system("cls || clear");
            menuCadastro();
            break;
        case 2:
            system("cls || clear");
                   venda();
```

```
            break;
        case 3:
            system("cls || clear");
            menuVenda();
            break;
        case 4:
            system("cls || clear");
            menuEstoque();
            break;
        default: defaultMessage(); break;
        }
    } while (iOp != 0);
}

int menuCaixa()
{
    int iOp;
    do
    {
         Funcionario *fAux = fFuncionarioInicial;
    system("cls || clear");

printf("================================================================================
================================================================================
================================================================================
========");
                                printf("\t   2vjmig\n\n");

       printf("\t\t\t\t*****************************************\n");
                           printf("\t\t\t\t\t\t   MENU\n");

       printf("\t\t\t\t*****************************************\n");
                       printf("   %s: ", cvCargoFunc);
                       printf("%s ", cvNFuncionario);
                           printf("\n\t\t\t\t\t\t\t\t\t\t\t\t\t\tiMerc.vs2018\n");

       printf("================================================================
================================================================================
================================================================================
===============\n");

       printf("0 - Fazer Logoff\n");
       printf("1 - Venda\n");
       printf("\n Escolha a opcao desejada: ");
       scanf("%d", &iOp);
       switch(iOp){
             case 0: VerificarLogin(); break;
             case 1: venda(); break;
             }
    } while (iOp != 0);
}

int menuCadastro()
{
    int iOp;
    do
    {
      tela("\t    MENU CADASTROS");
        printf("\n0- Voltar");
```

```
        printf("\n1- Cadastrar Funcionario");
        printf("\n2- Cadastrar Cliente");
        printf("\n3- Cadastrar Produto");
        printf("\n4- Cadastrar Fornecedor");
        printf("\n5 - Listar");
        printf("\n6 - Alterar");
        printf("\n7 - Remover");
        printf("\n\nEntre com a opcao desejada: ");
        scanf("%d", &iOp);
        switch (iOp)
        {
        case 0:
             system("cls || clear");
            break;
        case 1:
            cadastrarFuncionario();
            break;
        case 2:
            cadastrarCliente();
            break;
        case 3:
            cadastrarProduto();
            break;
        case 4:
            cadastrarFornecedor();
            break;
        case 5:
            menuListar();
            break;
        case 6:
            menuAlterar();
            break;
        case 7:
            menuRemover();
            break;
        default:
            defaultMessage();
            break;
        }
    } while (iOp != 0);
}

int menuListar(){
    int iOp;
    do{
        tela("\t    LISTAR CADASTROS");
        printf("\n0 - Voltar");
        printf("\n1- Listar Funcionario");
        printf("\n2- Listar Clientes");
        printf("\n3- Listar Produto");
        printf("\n4- Listar Fornecedor");
        printf("\nEntre com a opcao desejada: ");
        scanf("%d", &iOp);
        switch(iOp){
            case 0: system("cls || clear");break;
            case 1: listarFuncionario(); break;
            case 2: listarCliente();break;
            case 3: listarProdutos();break;
            case 4: listarFornecedor();break;
```

```
            default: defaultMessage();break;
        }
    }while(iOp != 0);
}

int menuAlterar()
{
    int iOp;
    do
    {
        tela("\t    ALTERAR CADASTROS");
        printf("\n0 - Voltar");
        printf("\n1- Alterar Funcionario");
        printf("\n2- Alterar Clientes");
        printf("\n3- Alterar Produto");
        printf("\n4- Alterar Fornecedor");
        printf("\nEntre com a opcao desejada: ");
        scanf("%d", &iOp);
        switch (iOp)
        {
        case 0:
            menuCadastro();
            break;
        case 1:
            alteraFuncionario();
            break;
        case 2:
            alteraCliente();
            break;
        case 3:
            alteraProduto();
            break;
        case 4:
            alteraFornecedor();
            break;

        default:
            defaultMessage();
            break;
        }

    } while (iOp < 0 || iOp > 4);
}

int menuRemover(){
      int iOp;

      do{
            tela("\t    REMOVER CADASTROS");
        printf("\n0 - Voltar");
        printf("\n1- Remover Funcionario");
        printf("\n2- Remover Clientes");
        printf("\n3- Remover Produto");
        printf("\n4- Remover Fornecedor");
        printf("\nEntre com a opcao desejada: ");
        scanf("%d", &iOp);
        switch(iOp){
             case 0:
                   menuCadastro();
```

```
                        break;
                case 1:
                        removeFuncionario();
                        break;
                case 2:
                        removeCliente();
                        break;
                case 3:
                        removeProduto();
                        break;
                case 4:
                        removeFornecedor();
                        break;
                default:
                        defaultMessage();
                        break;
                }
        }while(iOp < 0 || iOp > 4);
}


void menuVenda(){
        int iDia, iMes, iAno, iNaoEncontrado=0;
        int iOp;
        do{
                Venda *vAux = vVendaInicial;
                tela("\t    LISTA DE VENDAS");
                printf("0 - Voltar\n");
                printf("1 - Listar todas as vendas\n");
                printf("2 - Pesquisar Venda por Data\n");
                printf("3 - Pesquisar por dia\n");
                printf("4 - Pesquisar por mes");
                printf("Entre com a opcao desejada: ");
                scanf("%d", &iOp);
                switch(iOp){
                        case 0: system("cls || clear"); break;
                        case 1: listarVenda(); break;
                        case 2: pesqVendaData();break;
                        case 3:       system("cls || clear");
                        tela("\tLISTA DE VENDAS POR DIA");
                        if(vVendaInicial == NULL){
                                nullList();
                        }else{
                                printf("\nEntre com o dia desejado: ");
                                        scanf("%d", &iDia);
                                        while(vAux != NULL)
                                        {
                                                if(vAux->iDia == iDia)
                                                {
                                                        printf("Data: %d/%d/%d\t|\tCodigo
Venda: %d\t|\t Total da Venda: R$%.2f\n", vAux->iDia, vAux->iMes, vAux-
>iAno,vAux->iCodVenda, vAux->dTotalCompra);

        printf("\n==========================================================================
=============================================================\n");
                                                }else{
                                                        notFound("Venda");
                                                }
                                                vAux = vAux->vendaProximo;
```

```
                                    printf("\n\t\t\t\t   PRESSIONE QUALQUER TECLA
PARA VOLTAR!");
                                    }
                              }
                              fflush(stdin);
                               getchar();
                                    break;
                        case 4: tela("\tLISTA DE VENDAS POR MES");
                                    if(vVendaInicial == NULL){
                              nullList();
                        }else{
                              printf("\nEntre com o mes desejado: ");
                                    scanf("%d", &iMes);
                                    while(vAux != NULL)
                                    {
                                          if(vAux->iMes == iMes)
                                          {
                                                printf("Data: %d/%d/%d\t|\tCodigo
Venda: %d\t|\t Total da Venda: R$%.2f\n", vAux->iDia, vAux->iMes, vAux-
>iAno,vAux->iCodVenda, vAux->dTotalCompra);

      printf("\n=================================================================
============================================================\n");
                                          }else{
                                                notFound("Venda");
                                          }
                                          vAux = vAux->vendaProximo;
                                    printf("\n\t\t\t\t   PRESSIONE QUALQUER TECLA
PARA VOLTAR!");
                                    }
                              }
                              fflush(stdin);
                               getchar();
                                    break;
                  }
      }while(iOp != 0);
}


int menuEstoque(){
      int op;
      do{
                              tela("\t  CONTROLE DE ESTOQUE");
                              printf("\n0 - Voltar");
                        printf("\n1 - Produto(s) Acougue");
                        printf("\n2 - Produto(s) Adega e Bebidas");
                        printf("\n3 - Produto(s) Aves");
                        printf("\n4 - Produto(s) Bazar");
                        printf("\n5 - Produto(s) Frios e Laticinios");
                        printf("\n6 - Produto(s) Higiene e Limpeza");
                        printf("\n7 - Produto(s) Hortifruti");
                        printf("\n8 - Produto(s) Mercearia");
                        printf("\n9 - Produto(s) Padaria e Confeitaria");
                        printf("\n10 - Produto(s) Pescado\n");
                        printf("Entre com a opcao desejada: ");
                        scanf("%d",&op);
                        switch(op){
                              case 0: system("cls || clear"); break;
                              case 1: listarEstoque("Acougue"); break;
```

```
                        case 2: listarEstoque("Adega e Bebidas"); break;
                        case 3: listarEstoque("Aves"); break;
                                case 4: listarEstoque("Bazar"); break;
                                case 5: listarEstoque("Frios e Laticinios");
break;
                                case 6: listarEstoque("Higiene e Limpeza");
break;
                                case 7: listarEstoque("Hortifruti"); break;
                                case 8: listarEstoque("Mercearia"); break;
                                case 9: listarEstoque("Padaria e Confeitaria");
break;
                                case 10: listarEstoque("Pescado"); break;
                        default: defaultMessage();break;
                    }
            }while((op<0) || (op>10));
}

void tela(char tela[])
{
       system("cls || clear");

printf("================================================================================
================================================================================
================================================================================
========");
    printf("\t\t2vjmig\n\n");
    printf("\t\t\t\t**********************************************************\n");
    printf("\t\t\t\t\t%s\n", tela);
    printf("\t\t\t\t**********************************************************\n");
    printf("\n\t\t\t\t\t\t\t\t\t\t\t\t\tiMerc.vs2018\n");

printf("================================================================================
================================================================================
================================================================================
========\n");
}

void CadastroLogin(char cvCadastroLogin[], char cvCadastroSenha[])
{
    char cCaracter;
    char cvSenhaIgual[07];
    int iQtdCaracter;
    int iSenhaIgual = 0, z = 0;
    do
    {
        printf("\nDigite o login: ");
        fflush(stdin); //Limpando o buffer do teclado
        strcpy(cvCadastroLogin,Nome());
        printf("\nDigite a senha: ");
        iQtdCaracter = 0;
        do
        {
                    z=0;
            cCaracter = getch();
            if (isprint(cCaracter) && (iQtdCaracter != 6))
            {                                                   //Analisa se o valor
da vari?vel c ? imprimivel
                cvCadastroSenha[iQtdCaracter] = cCaracter; //Se for, armazena o
caractere
```

```
                iQtdCaracter++;
                printf("*"); //imprime o * Anterisco
            }
            else if (cCaracter == 8 && iQtdCaracter)
            { //8 ? o caractere BackSpace na tabela ASCII, && a analisa se a ?
diferente de 0
                cvCadastroSenha[iQtdCaracter] = '\0';
                iQtdCaracter--;
                printf("\b \b"); //Apagando o caractere digitado
            }else if(cCaracter==13 && iQtdCaracter!=0){
             z=1;
                    }
        } while (z != 1); //13 ? o valor de ENTER na tabela ASCII
        cvCadastroSenha[iQtdCaracter] = '\0';

        printf("\nConfirmacao da senha: ");
        iQtdCaracter = 0;
        z = 0;
        do
        {
            cCaracter = getch();
            if (isprint(cCaracter) && (iQtdCaracter != 6))
            { //Analisa se o valor de c ? imprim?vel
                cvSenhaIgual[iQtdCaracter] = cCaracter;
                iQtdCaracter++;
                printf("*"); //Imprimindo apenas o asterisco *
            }
            else if (cCaracter == 8 && iQtdCaracter)
            {
                cvSenhaIgual[iQtdCaracter] = '\0';
                iQtdCaracter--;
                printf("\b \b"); //Apagando os caracteres digitados
            }else if(cCaracter==13 && iQtdCaracter!=0){
             z=1;
                    }
        } while (z != 1); //13 ? o valor de ENTER na tabela ASCII
        cvSenhaIgual[iQtdCaracter] = '\0';

        if (!strcmp(cvCadastroSenha, cvSenhaIgual))
        {
            //strcmp retorna 0 se as vari?veis forem iguais. !strcmp ? o mesmo
que strcmp==0
            printf("\n\n\t\t\tCADASTRO EFETUADO COM SUCESSO...\n\n");
            iSenhaIgual = 1;
        }
        else
        {
            printf("\n\n\t\a\tSENHAS DIFERENTES - DIGITE NOVAMENTE...\n\n");
        }
    } while (iSenhaIgual != 1); //Enquanto b e d forem diferente de zero 0
}

void VerificarLogin()
{
    char cCaracter;
    char cTecla;
    char cvAcessoLogin[50];
    char cvAcessoSenha[07];
    int iQtdCaracter;
```

```
    int iSenhaIgual = 0, z = 0;
    char *t="SEJA BEM VINDO, FACA SEU LOGIN!!";
      tela(t);
      /****** 1 SOLUCAO PARA CASO NAO TENHA LOGIN ********/
      //verifica se tem Login
      if(!strcmp(cvAdm, "sim")){
    printf("\n\t\t\t\tPressione a tecla F2 para encerrar o programa ou
\n\t\t\t\tPressione qualquer outra para ir a tela de Login!!!");

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

printf("\n|||||||||||||||||||||||||||||||||||__||||__|||||||||||||||||||||||||
|||||||||||||||||||||||||||||||||||||||||||||||||||");
    printf("\n|||||||||||||||||||||||||||||||||  |||  | ||
|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||");
    printf("\n|||||||||||||||||||||||||||||  |||  |||| |||  |||||    ||||  ||
||||||     |||||||||||||||||||||||||||||||||||||||||||||||||");
    printf("\n|||||||||||||||||||||||||||||||||  ||||||||  ||||  ||  |||  | ||
|||  |||||||||||||||||||||||||||||||||||||||||||||||||");
    printf("\n|||||||||||||||||||||||||||  |||  ||||||||  |||  |TTTTT||
|||||||||  |||||||||||||||||||||||||||||||||||||||||||||||||");
    printf("\n|||||||||||||||||||||||||||  |||  ||||||||  |||  ||||||||
|||||||||  |||||||||||||||||||||||||||||||||||||||||||||||||");
    printf("\n|||||||||||||||||||||||||||  |||  ||||||||  |||||     |||
|||||||||      |||||||||||||||||||||||||||||||||||||||||||||||");

printf("\n|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
|||||||||||||||||||||||||||||||||||||||||||||||||||");
      cTecla = getch();
    if(cTecla == NULL){
      cTecla = getch();
      switch(cTecla){
      case 60:
            return 0;
            break;
            }
      }
      }else{
      tela("\tBEM VINDO VINDO");
      printf("\n\t\tPressione F2 para sair ou qualquer outra tecla para
proseguir com seu cadastro!!");

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

printf("\n|||||||||||||||||||||||||||||||||||__||||__|||||||||||||||||||||||||
|||||||||||||||||||||||||||||||||||||||||||||||||||");
    printf("\n|||||||||||||||||||||||||||||||||  |||  | ||
|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||");
    printf("\n|||||||||||||||||||||||||||||  |||  |||| |||  |||||    ||||  ||
||||||     |||||||||||||||||||||||||||||||||||||||||||||||||");
    printf("\n|||||||||||||||||||||||||||||||||  ||||||||  ||||  ||  |||  | ||
|||  |||||||||||||||||||||||||||||||||||||||||||||||||");
    printf("\n|||||||||||||||||||||||||||  |||  ||||||||  |||  |TTTTT||
|||||||||  |||||||||||||||||||||||||||||||||||||||||||||||||");
    printf("\n|||||||||||||||||||||||||||  |||  ||||||||  |||  ||||||||
|||||||||  |||||||||||||||||||||||||||||||||||||||||||||||||");
    printf("\n|||||||||||||||||||||||||||  |||  ||||||||  |||||     |||
```

```
|||||||||      ||||||||||||||||||||||||||||||||||||||||||||");

printf("\n||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
||||||||||||||||||||||||||||||||||||||||||||||");
       cTecla = getch();
    if(cTecla == NULL){
       cTecla = getch();
       switch(cTecla){
       case 60:
             return 0;
             break;
             }
       getchar();
       }
       cadastrarFuncionario();
}
       t="SEJA BEM VINDO, FACA SEU LOGIN!!";
       tela(t);
       do
    {

        printf("\nEntre com o login: ");
        fflush(stdin); //Limpando o buffer do teclado
        strcpy(cvAcessoLogin,Nome());
        printf("\nEntre com a senha: ");
        iQtdCaracter = 0;
        do
        {
            cCaracter = getch();
            if (isprint(cCaracter) && (iQtdCaracter != 6))
            { //Analisa se o valor de c ? imprim?vel
                cvAcessoSenha[iQtdCaracter] = cCaracter;
                iQtdCaracter++;
                printf("*"); //Imprimindo apenas o asterisco *
            }
            else if (cCaracter == 8 && iQtdCaracter)
            {
                cvAcessoSenha[iQtdCaracter] = '\0';
                iQtdCaracter--;
                printf("\b \b"); //Apagando os caracteres digitados
            }else if(cCaracter==13 && iQtdCaracter!=0){
             z=1;
                    }
        } while (z != 1); //13 ? o valor de ENTER na tabela ASCII
        cvAcessoSenha[iQtdCaracter] = '\0';

        Funcionario *fAux = fFuncionarioInicial;
        while (fAux != NULL)
        {
            if( (!strcmp(fAux->cvLogin, cvAcessoLogin)) && (!strcmp(fAux-
>cvSenha,cvAcessoSenha))){
                if(!strcmp(fAux->cvCargo, "ADMINISTRADOR")){
                      iSenhaIgual = 1;
                      strcpy(cvNFuncionario, fAux->cvNomeFunc);
                      strcpy(cvCargoFunc, fAux->cvCargo);
                      menu();
                          }else if(!strcmp(fAux->cvCargo, "CAIXA")){
                      iSenhaIgual = 1;
                      strcpy(cvNFuncionario, fAux->cvNomeFunc);
```

```
                        strcpy(cvCargoFunc, fAux->cvCargo);
                        menuCaixa();
                            }else if(!strcmp(fAux->cvCargo, "FINANCEIRO")){
                        iSenhaIgual = 1;
                        strcpy(cvNFuncionario, fAux->cvNomeFunc);
                        strcpy(cvCargoFunc, fAux->cvCargo);
                        menuEstoque();
                            }
                }
                fAux = fAux->funProximo;
            }

            if(iSenhaIgual == 0){
                 z=0;
                 printf("\n\n\t\t\tUSUARIO NAO CADASTRADO... TENTE NOVAMENTE!
\n\n");
                 }

        } while (iSenhaIgual != 1);
}

int cadastrarProduto(){
       int op;
         tela("\tCADASTRO DE PRODUTO");
    char cEscolha;
    do{
         Produto *pNovoProduto = (Produto*) malloc(sizeof(Produto));
         pNovoProduto->pProximo = NULL;
              fflush(stdin);
              printf("\nCodigo do Produto: ");
         strcpy(pNovoProduto->cvCodigo,Nome());
         printf("\nNome do produto: ");
         strcpy(pNovoProduto->cvNome,Nome());
         fflush(stdin);
         printf("\nData de Validade: ");
         strcpy(pNovoProduto->cvValidade, FormatarData());
         pNovoProduto->cvValidade[strcspn(pNovoProduto->cvValidade, "\n")] =
'\0';
         fflush(stdin);
         printf("\nValor do Produto: ");
         pNovoProduto->fValor = pegaValorProd();
         printf("\nQuantidade no Estoque(APENAS NUMEROS): ");
         scanf("%d", &pNovoProduto->iQtdEstoque);
         printf("\nEstoque de Seguranca(APENAS NUMEROS): ");
         scanf("%d", &pNovoProduto->iEstoqSeguranca);
         printf("\nCodigo do Fornecedor(APENAS NUMEROS): ");
         scanf("%d", &pNovoProduto->iCodigoFornecedor);
         fflush(stdin);
         do{
                              printf("\Setor Produto: ");
                        printf("\n1: Acougue");
                        printf("\n2: Adega e Bebidas");
                        printf("\n3: Aves");
                        printf("\n4: Bazar");
                        printf("\n5: Frios e Laticinios");
                        printf("\n6: Higiene e Limpeza");
                        printf("\n7: Hortifruti");
                        printf("\n8: Mercearia");
                        printf("\n9: Padaria e Confeitaria");
```

```
                    printf("\n10: Pescado");
                    printf("\nEntre com a opcao desejada: ");
                    scanf("%d",&op);
                    switch(op){
                         case 1:
                                    strcpy(pNovoProduto->cvSetor, "Acougue");
                               pNovoProduto->cvSetor[strcspn(pNovoProduto-
>cvSetor, "\n")] = '\0';
                               break;
                         case 2:
                                    strcpy(pNovoProduto->cvSetor, "Adega e
Bebidas");
                               pNovoProduto->cvSetor[strcspn(pNovoProduto-
>cvSetor, "\n")] = '\0';
                               break;
                         case 3:
                                    strcpy(pNovoProduto->cvSetor, "Aves");
                               pNovoProduto->cvSetor[strcspn(pNovoProduto-
>cvSetor, "\n")] = '\0';
                               break;
                               case 4:
                                    strcpy(pNovoProduto->cvSetor, "Bazar");
                               pNovoProduto->cvSetor[strcspn(pNovoProduto-
>cvSetor, "\n")] = '\0';
                               break;
                               case 5:
                                    strcpy(pNovoProduto->cvSetor, "Frios e
Laticinios");
                               pNovoProduto->cvSetor[strcspn(pNovoProduto-
>cvSetor, "\n")] = '\0';
                               break;
                               case 6:
                                    strcpy(pNovoProduto->cvSetor, "Higiene e
Limpeza");
                               pNovoProduto->cvSetor[strcspn(pNovoProduto-
>cvSetor, "\n")] = '\0';
                               break;
                               case 7:
                                    strcpy(pNovoProduto->cvSetor,
"Hortifruti");
                               pNovoProduto->cvSetor[strcspn(pNovoProduto-
>cvSetor, "\n")] = '\0';
                               break;
                               case 8:
                                    strcpy(pNovoProduto->cvSetor,
"Mercearia");
                               pNovoProduto->cvSetor[strcspn(pNovoProduto-
>cvSetor, "\n")] = '\0';
                               break;
                               case 9:
                                    strcpy(pNovoProduto->cvSetor, "Padaria e
Confeitaria");
                               pNovoProduto->cvSetor[strcspn(pNovoProduto-
>cvSetor, "\n")] = '\0';
                               break;
                               case 10:
                                    strcpy(pNovoProduto->cvSetor, "Pescado");
                               pNovoProduto->cvSetor[strcspn(pNovoProduto-
>cvSetor, "\n")] = '\0';
```

```
                                    break;
                            default:
                                    defaultMessage();
                                    fflush(stdin);
                    break;
                        }
                }while((op<1) || (op>10));

            if(pProdutoInicial == NULL){
                pProdutoInicial = pNovoProduto;
            }else{
                Produto *pAux = pProdutoInicial;
                while(pAux->pProximo != NULL){
                    pAux = pAux->pProximo;
                }
                pAux->pProximo = pNovoProduto;
            }


                cEscolha = escolhaRegistro();

            system("cls || clear");
            tela("\tCADASTRO DE PRODUTO");
        }while(cEscolha != 'n');
        gravarProduto();
}
float pegaValorProd(){
       char cCaracter;
       int iMax=100;
    char cValor[iMax],c[iMax];
    int iQtdCaracter=0, x=0, z, w;
    char cPonto = 46;
    float fValor = 0;

            do
            {
                fflush(stdin);
                if(x<iMax){
                        c[x]='\b';
                        x++;
                        }
                cCaracter = getch();
                if (isprint(cCaracter))
                { //Analisa se o valor de c ? imprim?vel
                if(cCaracter=='0'|| cCaracter=='1'|| cCaracter=='2'||cCaracter=='3'
|| cCaracter=='4'||cCaracter=='5' || cCaracter=='6'||cCaracter=='7' ||
cCaracter=='8'||cCaracter=='9'){
                    iQtdCaracter++;
                    if(iQtdCaracter==1){
                        cValor[0] = 48;
                        cValor[1]=46;
                        cValor[2]=48;
                        cValor[3]=cCaracter;
                    printf("%s",cValor); //Imprimindo valor
                        }else if(iQtdCaracter==2){
                                printf("\b\b\b\b");
                                        cValor[0] = 48;
                                        cValor[1] = 46;
                        cValor[2]=cValor[3];
```

```
                    cValor[3]=cCaracter;
                printf("%s",cValor); //Imprimindo valor*
                    }else if(iQtdCaracter==3){
                           printf("\b\b\b\b");
                                  cValor[0] = cValor[2];
                                  cValor[1] = 46;
                    cValor[2]=cValor[3];
                    cValor[3]=cCaracter;
                printf("%s",cValor); //Imprimindo valor *
        }else if(iQtdCaracter>3){
             z=iQtdCaracter-2;
             w=iQtdCaracter-3;
                    printf("%s",c);
                    //numero antes do ponto final
                    cValor[w]=cValor[z];
                    //ponto final
                    cValor[z]=46;
                    cValor[x]=cCaracter;
            printf("%s",cValor); //Imprimindo valor *
                    }
             }
        }else if (cCaracter == 8 && iQtdCaracter)
            {
                cValor[iQtdCaracter] = '\0';
                iQtdCaracter--;
                x--;
                printf("\b \b"); //Apagando os caracteres digitados
                    }
        } while (cCaracter != 13); //13 ? o valor de ENTER na tabela ASCII
        iQtdCaracter+=1;
        cValor[iQtdCaracter] = '\0';
        fValor = atof(cValor);
        return fValor;
}

int FormatarData(){
  char cCaracter;
    char data[10];
    int iQtdCaracter,x, z=0;
        iQtdCaracter = 0;
        do
        {
             fflush(stdin);
            cCaracter = getch();
            if (isprint(cCaracter) && (iQtdCaracter != 10))
            { //Analisa se o valor de c ? imprim?vel
            if(cCaracter=='0'|| cCaracter=='1'|| cCaracter=='2'||cCaracter=='3'
|| cCaracter=='4'||cCaracter=='5' || cCaracter=='6'||cCaracter=='7' ||
cCaracter=='8'||cCaracter=='9'){
                data[iQtdCaracter] = cCaracter;
                iQtdCaracter++;
                x++;
                printf("%c",cCaracter);
                if(iQtdCaracter==2){
                    printf("\b\b");
                    data[2] = '/';
                    printf("%s",data);
                    iQtdCaracter++;
                           }else if(iQtdCaracter==5){
```

```
                        printf("\b\b\b\b\b");
                    data[5] = '/';
                    printf("%s",data);
                    iQtdCaracter++;
                    }
                    }
             }
            else if (cCaracter == 8 && iQtdCaracter)
            {
                data[iQtdCaracter] = '\0';
                iQtdCaracter--;
                printf("\b \b"); //Apagando os caracteres digitados
                    }else if(cCaracter==13 && iQtdCaracter==10){
                        z=1;
                    }
        } while (z != 1); //13 ? o valor de ENTER na tabela ASCII
        data[iQtdCaracter] = '\0';
       return data;
}

int FormatarTelefone(){
  char cCaracter;
    char fTelefone[13];
    int iQtdCaracter,x, z=0;
        iQtdCaracter = 1;
        printf("(");
        do
        {
             fflush(stdin);
            cCaracter = getch();
            if (isprint(cCaracter) && (iQtdCaracter != 13))
            { //Analisa se o valor de c ? imprim?vel
            if(cCaracter=='0'|| cCaracter=='1'|| cCaracter=='2'||cCaracter=='3'
|| cCaracter=='4'||cCaracter=='5' || cCaracter=='6'||cCaracter=='7' ||
cCaracter=='8'||cCaracter=='9'){
                        fTelefone[iQtdCaracter] = cCaracter;
                iQtdCaracter++;
                fTelefone[0] ='(';
                x++;
                printf("%c",cCaracter);
                if(iQtdCaracter==3){
                    printf("\b\b\b");
                    fTelefone[3] = ')';
                    printf("%s",fTelefone);
                    iQtdCaracter++;
                        }else if(iQtdCaracter==8){
                        printf("\b\b\b\b\b\b\b\b");
                    fTelefone[8] = '-';

       printf("%c%c%c%c%c%c%c%c%c",fTelefone[0],fTelefone[1],fTelefone[2],fTelef
one[3],fTelefone[4],fTelefone[5],fTelefone[6],fTelefone[7],fTelefone[8],fTelefon
e[9]);
                    iQtdCaracter++;
                    }
                    }
             }
            else if (cCaracter == 8 && iQtdCaracter!=1)
            {fTelefone[0] = '\0';
                fTelefone[iQtdCaracter] = '\0';
```

```
                iQtdCaracter--;
                printf("\b \b"); //Apagando os caracteres digitados
                    }else if(cCaracter==13 && iQtdCaracter==13){
                          z=1;
                    }
        } while (z!=1); //13 ? o valor de ENTER na tabela ASCII
        fTelefone[iQtdCaracter] = '\0';
       return fTelefone;
}

int FormatarCelular(){
  char cCaracter;
    char cCelular[14];
    int iQtdCaracter,x, z=0;
        iQtdCaracter = 1;
        printf("(");
        do
        {
             fflush(stdin);
            cCaracter = getch();
            if (isprint(cCaracter) && (iQtdCaracter != 14))
            { //Analisa se o valor de c ? imprim?vel
            if(cCaracter=='0'|| cCaracter=='1'|| cCaracter=='2'||cCaracter=='3'
|| cCaracter=='4'||cCaracter=='5' || cCaracter=='6'||cCaracter=='7' ||
cCaracter=='8'||cCaracter=='9'){
                          cCelular[iQtdCaracter] = cCaracter;
                iQtdCaracter++;
                cCelular[0] ='(';
                x++;
                printf("%c",cCaracter);
                if(iQtdCaracter==3){
                    printf("\b\b\b");
                    cCelular[3] = ')';
                    printf("%s",cCelular);
                    iQtdCaracter++;
                          }else if(iQtdCaracter==9){
                          printf("\b\b\b\b\b\b\b\b\b");
                    cCelular[9] = '-';

       printf("%c%c%c%c%c%c%c%c%c%c",cCelular[0],cCelular[1],cCelular[2],cCelula
r[3],cCelular[4],cCelular[5],cCelular[6],cCelular[7],cCelular[8],cCelular[9],cCe
lular[10]);
                    iQtdCaracter++;
                    }
                    }
             }
            else if (cCaracter == 8 && iQtdCaracter!=1)
            {cCelular[0] = '\0';
                cCelular[iQtdCaracter] = '\0';
                iQtdCaracter--;
                printf("\b \b"); //Apagando os caracteres digitados
                    }else if(cCaracter==13 && iQtdCaracter==14){
                          z=1;
                    }
        } while (z != 1); //13 ? o valor de ENTER na tabela ASCII
        cCelular[iQtdCaracter] = '\0';
        fflush(stdin);
       return cCelular;
}
```

```
int FormatarCPF(){
  char cCaracter;
    char cCpf[14];
    int iQtdCaracter,x, z=0;
        iQtdCaracter = 0;
        do
        {
             fflush(stdin);
            cCaracter = getch();
            if (isprint(cCaracter) && (iQtdCaracter != 14))
            { //Analisa se o valor de c ? imprim?vel
            if(cCaracter=='0'|| cCaracter=='1'|| cCaracter=='2'||cCaracter=='3'
|| cCaracter=='4'||cCaracter=='5' || cCaracter=='6'||cCaracter=='7' ||
cCaracter=='8'||cCaracter=='9'){
                cCpf[iQtdCaracter] = cCaracter;
                iQtdCaracter++;
                x++;
                printf("%c",cCaracter); //Imprimindo
                if(iQtdCaracter==3){
                    printf("\b\b\b");
                    cCpf[3] = '.';
                    printf("%s",cCpf);
                    iQtdCaracter++;
                          }else if(iQtdCaracter==7){
                          printf("\b\b\b\b\b\b\b");
                    cCpf[7] = '.';
                    printf("%s",cCpf);
                    iQtdCaracter++;
                    }else if(iQtdCaracter==11){
                          printf("\b\b\b\b\b\b\b\b\b\b\b\b");
                    cCpf[11] = '-';
                    printf("%s",cCpf);
                    iQtdCaracter++;
                    }
                    }
             }
            else if (cCaracter == 8 && iQtdCaracter)
            {
                cCpf[iQtdCaracter] = '\0';
                iQtdCaracter--;
                printf("\b \b"); //Apagando os caracteres digitados
                    }else if(cCaracter==13 && iQtdCaracter==14){
                          z=1;
                    }
        } while (z != 1); //13 ? o valor de ENTER na tabela ASCII
        cCpf[iQtdCaracter] = '\0';
       return cCpf;
}

int Nome(){
  char cCaracter;
    char cNome[200];
    int iQtdCaracter,x, z=0;
        iQtdCaracter = 0;
        do
        {
             fflush(stdin);
            cCaracter = getch();
```

```
            if (isprint(cCaracter))
            { //Analisa se o valor de c ? imprim?vel
            cNome[iQtdCaracter] = cCaracter;
                iQtdCaracter++;
                printf("%c",cCaracter); //Imprimindo
             }
            else if (cCaracter == 8 && iQtdCaracter)
            {
                cNome[iQtdCaracter] = '\0';
                iQtdCaracter--;
                printf("\b \b"); //Apagando os caracteres digitados
                   }else if(cCaracter==13 && iQtdCaracter!=0){
                         z=1;
                   }
        } while (z != 1); //13 ? o valor de ENTER na tabela ASCII
        cNome[iQtdCaracter] = '\0';
       return cNome;
}

void listarProdutos(){
       //variaveis auxiliares
       char c[200]="                                                    ";
       char a[200],b[200];
       int iA;
    tela("\tLISTA DE PRODUTOS");
    if(pProdutoInicial == NULL){
         nullList();
    }else{
        Produto *pAux = pProdutoInicial;
        while(pAux != NULL){
             //nome produto
             iA=54 - strlen(pAux->cvNome);
       strncat(pAux->cvNome,c,iA);
        sprintf(a, "%d", pAux->iQtdEstoque );
       iA=14 - strlen(a);
       strncat(a,c,iA);
       iA=14 - strlen(pAux->cvCodigo);
       strncat(pAux->cvCodigo,c,iA);
       sprintf(b, "%d", pAux->iEstoqSeguranca );
       iA=13 - strlen(b);
       strncat(b,c,iA);
            printf("Nome: %s| Validade: %s   | Valor: %.2f", pAux->cvNome, pAux-
>cvValidade,pAux->fValor);
            printf("\n-------------------------------------------------------------
-------------------------------------------------------------");
                  printf("\nQuantidade Est: %s| Est.Seguranca: %s| Codigo: %s
| Cod.Fornecedor: %d",a ,b,pAux->cvCodigo, pAux->iCodigoFornecedor);

printf("\n=============================================================================
================================================\n");

            pAux = pAux->pProximo;

        }
    }
    printf("\n\t\t\t\t   PRESSIONE QUALQUER TECLA PARA VOLTAR!");
    fflush(stdin);
    getchar();
}
```

```
int alteraProduto()
{
    int iOp;

       do{
       tela("\tALTERAR PRODUTO");
       printf("0 - Sair\n");
       printf("1 - Alterar Codigo\n");
       printf("2 - Alterar Nome\n");
       printf("3 - Alterar Valor\n");
       printf("4 - Alterar Quant. Estoque\n");
       printf("5 - Alterar Estoque Seguranca\n");
       printf("Escolha uma opcao: ");
       scanf("%d", &iOp);
       switch(iOp){
              case 0:
                     menuAlterar();
                     break;
              case 1:
                     system("cls || clear");
                     alteraCodigoProduto();
                     break;
              case 2:
                     system("cls || clear");
                     alteraNomeProduto();
                     break;
              case 3:
                     system("cls || clear");
                     alteraValorProduto();
                     break;
              case 4:
                     system("cls || clear");
                     alteraQtdProduto();
                     break;
              case 5:
                           system("cls || clear");
                           alteraEstoqueSeg();
                     break;
              default:
                     defaultMessage();
                     break;
              }
       }while(iOp != 0);
}

int alteraCodigoProduto(){
       Produto *pAux = pProdutoInicial;
       char cvCodigoP[51];
       char cvNovoCod[51];
       int iNaoEncontrado = 0;

    fflush(stdin);
    printf("\nEntre com o codigo do produto a ser alterado: ");
    strcpy(cvCodigoP,Nome());


       while(pAux != NULL){
              if(!strcmp(pAux->cvCodigo, cvCodigoP)){
```

```
                        printf("Codigo atual: %s\tProduto: %s\n", pAux->cvCodigo,
pAux->cvNome);
                        printf("Digite o novo codigo: ");
                        strcpy(cvNovoCod,Nome());
                        strcpy(pAux->cvCodigo, cvNovoCod);
                        iNaoEncontrado++;
                }
                pAux = pAux->pProximo;
        }

        if(!iNaoEncontrado){
                notFound("Produto");
        }else{
                alteradoSucesso("O codigo do produto");
        }
}

int alteraNomeProduto(){
        Produto *pAux = pProdutoInicial;
        char cvCodigoP[51];
        char cvNovoNome[201];
        int iNaoEncontrado = 0;

        fflush(stdin);
    printf("\nEntre com o codigo do produto a ser alterado: ");
    strcpy(cvCodigoP,Nome());

        while(pAux != NULL){
                if(!strcmp(pAux->cvCodigo, cvCodigoP)){
                        printf("Nome atual: %s\n", pAux->cvNome);
                        printf("Digite o novo nome: ");
                        strcpy(cvNovoNome,Nome());
                        strcpy(pAux->cvNome,cvNovoNome);
                        iNaoEncontrado++;
                }
                pAux = pAux->pProximo;
        }
        if(!iNaoEncontrado){
                notFound("Produto");
        }else{
                alteradoSucesso("O nome do produto");
        }
}

int alteraValorProduto(){
        Produto *pAux = pProdutoInicial;
        char cvCodigoP[51];
        int iNaoEncontrado = 0;
        float fNovoValor;

        fflush(stdin);
    printf("\nEntre com o codigo do produto a ser alterado: ");
    strcpy(cvCodigoP,Nome());

        while(pAux != NULL){
                if(!strcmp(pAux->cvCodigo, cvCodigoP)){
                        printf("Valor atual: %.2f\tProduto: %s\n", pAux->fValor,
pAux->cvNome);
                        printf("Digite o novo valor: R$ ");
```

```
                    pAux->fValor = pegaValorProd();
                    iNaoEncontrado++;
                }
                pAux = pAux->pProximo;
        }
        if(!iNaoEncontrado){
                notFound("Produto");
        }else{
                alteradoSucesso("O valor do produto");
        }
}

int alteraQtdProduto(){
        Produto *pAux = pProdutoInicial;
        char cvCodigoP[51];
        int iNaoEncontrado = 0;
        int iNovoQtd;

        fflush(stdin);
    printf("\nEntre com o codigo do produto a ser alterado: ");
    strcpy(cvCodigoP,Nome());

        while(pAux != NULL){
                if(!strcmp(pAux->cvCodigo, cvCodigoP)){
                    printf("Qtd. Estoque atual: %d\tProduto: %s\n", pAux-
>iQtdEstoque, pAux->cvNome);
                    printf("Digite a nova quantidade: ");
                    scanf("%d", &iNovoQtd);
                    pAux->iQtdEstoque += iNovoQtd;
                    iNaoEncontrado++;
                }
                pAux = pAux->pProximo;
        }
        if(!iNaoEncontrado){
                notFound("Produto");
        }else{
                alteradoSucesso("A quantidade do produto");
        }
}

int alteraEstoqueSeg(){
        Produto *pAux = pProdutoInicial;
        char cvCodigoP[51];
        int iNaoEncontrado = 0;
        int iNovoQtd;

        fflush(stdin);
    printf("\nEntre com o codigo do produto a ser alterado: ");
    strcpy(cvCodigoP,Nome());

        while(pAux != NULL){
                if(!strcmp(pAux->cvCodigo, cvCodigoP)){
                    printf("Qtd. Estoque Seguranca atual: %d\tProduto: %s\n",
pAux->iEstoqSeguranca, pAux->cvNome);
                    printf("Digite a nova quantidade de seguranca: ");
                    scanf("%d", &iNovoQtd);
                    pAux->iEstoqSeguranca += iNovoQtd;
                    iNaoEncontrado++;
                }
```

```
                pAux = pAux->pProximo;
        }
        if(!iNaoEncontrado){
                notFound("Produto");
        }else{
                alteradoSucesso("A quantidade do produto");
        }
}

void removeProduto()
{
        tela("REMOVER PRODUTO");
    Produto *pAux = pProdutoInicial;
    Produto *pAnterior;
    if (pProdutoInicial == NULL)
    {
        printf("\nA lista ja esta vazia!");
    }
    else
    {
        char cvCodigoP[51];
        char cvNomeTemp[200]; //Armazena o nome do item excluido temporariamente
        char cEscolha;
        int iNaoEncontrado = 0;
        printf("Entre com o codigo do produto que deseja remover: ");
        fflush(stdin);
        strcpy(cvCodigoP,Nome());

        while (pAux != NULL)
        {
            if (!strcmp(pAux->cvCodigo, cvCodigoP))
            {
                if (pProdutoInicial->pProximo == NULL)
                    pProdutoInicial = NULL; // se o inicial nao tiver proximo,
deve receber nulo
                strcpy(cvNomeTemp, pAux->cvNome);
                if (pAux == pProdutoInicial)
                {
                    pAux = pAux->pProximo;
                    free(pProdutoInicial);
                    pProdutoInicial = NULL;
                    pProdutoInicial = pAux;
                    removidoSucesso("Produto", cvNomeTemp);
                }
                else
                {
                    pAnterior->pProximo = pAux->pProximo;
                    free(pAux);
                    pAux = NULL;
                    removidoSucesso("Produto", cvNomeTemp);
                    return 0;
                }
                iNaoEncontrado++;
            }
            pAnterior = pAux;
            pAux = pAux->pProximo;
        }
        if (iNaoEncontrado == 0)
            notFound("Produto");
```

```
    }
}

int cadastrarFornecedor(){
    char cEscolha;
    tela("\tCADASTRO DE FORNECEDOR");
    do{
        Fornecedor *novoFornecedor = (Fornecedor*) malloc(sizeof(Fornecedor));
        novoFornecedor->fProximo = NULL;

        printf("\nNome Fornecedor: ");
        fflush(stdin);
        strcpy(novoFornecedor->cvNome,Nome());
        printf("\nNome Fantasia: ");
        strcpy(novoFornecedor->cvNomeFantasia,Nome());
        printf("\nEndereco: ");
        strcpy(novoFornecedor->cvEndereco,Nome());
        printf("\nCNPJ: ");
        strcpy(novoFornecedor->cvCnpj,Nome());
        printf("\nEmail: ");
        strcpy(novoFornecedor->cvEmail,Nome());
        printf("\nTelefone: ");
        strcpy(novoFornecedor->cvTelefone, FormatarTelefone());
        fflush(stdin);
        printf("\nCelular: ");
        strcpy(novoFornecedor->cvCelular, FormatarCelular());
        printf("\nCodigo Produto(APENAS NUMEROS): ");
        scanf("%d", &novoFornecedor->iCodigoProduto);

        if(fFornecedorInicial == NULL) {
            fFornecedorInicial = novoFornecedor;
        }else{
            Fornecedor *fAux = fFornecedorInicial;
            while(fAux->fProximo != NULL){
                fAux = fAux->fProximo;
            }
            fAux->fProximo = novoFornecedor;
        }

        cEscolha = escolhaRegistro();
        system("cls || clear");
        tela("\tCADASTRO DE FORNECEDOR");
    }while(cEscolha != 'n');
    gravarFornecedor();
}

void listarFornecedor(){
      //variaveis auxiliares
      char c[200]="                                                    ";
      int iA;
    tela("\tLISTA DE FORNECEDORES");
    if(fFornecedorInicial == NULL){
        nullList();
    }else{
        Fornecedor *fAux = fFornecedorInicial;
        while(fAux != NULL){
             iA=39 - strlen(fAux->cvNome);
      strncat(fAux->cvNome,c,iA);
      iA=39 - strlen(fAux->cvNomeFantasia);
```

```
       strncat(fAux->cvNomeFantasia,c,iA);
            iA=49 - strlen(fAux->cvEmail);
       strncat(fAux->cvEmail,c,iA);
                    printf("\n-----------------------------------------------------
-------------------------------------------------------------------");
            printf("\nNome: %s| Empresa: %s | CNPJ: %s", fAux->cvNome, fAux-
>cvNomeFantasia, fAux->cvCnpj);
            printf("\n-----------------------------------------------------------
-------------------------------------------------------------");
            printf("\nCelular: %s                                    |\tTelefone:
%s", fAux->cvCelular, fAux->cvTelefone);
            printf("\n-----------------------------------------------------------
-------------------------------------------------------------");
            printf("\nEmail: %s|\tEndereco: %s", fAux->cvEmail, fAux-
>cvEndereco);

printf("\n=============================================================================
=================================================\n");
            fAux = fAux->fProximo;
        }
        printf("\n\t\t\t\t   PRESSIONE QUALQUER TECLA PARA VOLTAR!");
    }
    char cListar;
    cListar = getchar();
    getchar();
}

void alteraFornecedor()
{
       tela("\tALTERAR FORNECEDOR");
    Fornecedor *fAux = fFornecedorInicial;
    char cvExNome[201];
    fflush(stdin);
    editMessage("fornecedor");
    strcpy(cvExNome,Nome());
    int iNaoEncontrado = 0;
    while (fAux != NULL)
    {
        if (!strcmp(fAux->cvNome, cvExNome) || !strcmp(fAux->cvNomeFantasia,
cvExNome))
        {
            fflush(stdin);
            printf("\nNome do Fornecedor: ");
            strcpy(fAux->cvNome,Nome());
            printf("\nNome Fantasia: ");
            strcpy(fAux->cvNomeFantasia,Nome());
            printf("\nEndereco: ");
            strcpy(fAux->cvEndereco,Nome());
            printf("\CNPJ: ");
            strcpy(fAux->cvCnpj,Nome());
            printf("\nEmail: ");
            strcpy(fAux->cvEmail,Nome());
            printf("\nTelefone: ");
        strcpy(fAux->cvTelefone, FormatarTelefone());
        fflush(stdin);
        printf("\nCelular: ");
        strcpy(fAux->cvCelular, FormatarCelular());
            printf("\nCodigo Produto(APENAS NUMEROS): ");
            scanf("%d", &fAux->iCodigoProduto);
```

```
            iNaoEncontrado++;
            break;
        }
        fAux = fAux->fProximo;
    }

    if (iNaoEncontrado == 0)
    {
        nullList();
    }
}

void removeFornecedor(){
    tela("\tREMOVER FORNECEDOR");
    Fornecedor *fAux = fFornecedorInicial;
    Fornecedor *fAnterior;
    if(fFornecedorInicial == NULL){
        nullDelete();
    }else{
            char cvNomeP[201];
            char cvNomeTemp[200]; //Armazena o nome do item excluido
temporariamente
            int iNaoEncontrado = 0;
        printf("Entre com o nome do fornecedor que deseja remover: ");
        fflush(stdin);
        strcpy(cvNomeP,Nome());

        while(fAux != NULL){
            if(!strcmp(fAux->cvNome, cvNomeP) || !strcmp(fAux->cvNomeFantasia,
cvNomeP)){
                    if(fFornecedorInicial->fProximo == NULL) fFornecedorInicial
= NULL;
                    strcpy(cvNomeTemp, fAux->cvNome);
                    if(fAux == fFornecedorInicial){
                           fAux = fAux->fProximo;
                           free(fFornecedorInicial);
                           fFornecedorInicial = NULL;
                           fFornecedorInicial = fAux;
                           removidoSucesso("Fornecedor", cvNomeTemp);
                           }else{
                                  fAnterior->fProximo = fAux->fProximo;
                                  free(fAux);
                                  fAux = NULL;
                                  removidoSucesso("Fornecedor", cvNomeTemp);
                                  return 0;
                           }
                           iNaoEncontrado++;
                    }
                    fAnterior = fAux;
                    fAux = fAux->fProximo;
            }
            if(iNaoEncontrado == 0 ) notFound("Fornecedor");

    }
}

int cadastrarCliente(){
    char cEscolha;
    tela("\tCADASTRO DE CLIENTE");
```

```
    do{
        Cliente *cNovocliente = (Cliente*) malloc(sizeof(Cliente));
        cNovocliente->cProximo = NULL;

        fflush(stdin);
        printf("\nNome: ");
        strcpy(cNovocliente->cvNomeCli,Nome());
        printf("\nCPF: ");
        fflush(stdin);
        strcpy(cNovocliente->cvCpf,FormatarCPF());
        fflush(stdin);
        printf("\nData de Nascimento: ");
        strcpy(cNovocliente->cvData_nascimento,FormatarData());
        printf("\nEmail: ");
        strcpy(cNovocliente->cvEmail,Nome());
        printf("\nTelefone: ");
        strcpy(cNovocliente->cvTelefone,FormatarTelefone());
        printf("\nCelular: ");
        strcpy(cNovocliente->cvCelular,FormatarCelular());
        printf("\nEndereco: ");
        strcpy(cNovocliente->cvEndereco,Nome());

        if(cClienteInicial == NULL){
            cClienteInicial = cNovocliente;
        }else{
            Cliente *cAux = cClienteInicial;
            while(cAux->cProximo != NULL){
                cAux = cAux->cProximo;
            }
            cAux->cProximo = cNovocliente;
        }
        cEscolha = escolhaRegistro();
        system("cls || clear");
        tela("\tCADASTRO DE CLIENTE");
    }while(cEscolha != 'n');
    gravarCliente();
}

void listarCliente(){
      //variaveis auxiliares
      char c[200]="                                                    ";
      int iA;
    tela("\tLISTA DE CLIENTES");
    if(cClienteInicial == NULL){
        nullList();
    }else{

        Cliente *cAux = cClienteInicial;
        while(cAux != NULL){
      iA=49 - strlen(cAux->cvEmail);
      strncat(cAux->cvEmail,c,iA);
                  printf("\n-------------------------------------------------
-------------------------------------------------------------------");
            printf("Nome: %s", cAux->cvNomeCli);
            printf("\n-------------------------------------------------------------
-------------------------------------------------------------");
            printf("\nCPF: %s                                        |\tData de
Nascimento: %s", cAux->cvCpf, cAux->cvData_nascimento);
            printf("\n-------------------------------------------------------------
```

```
------------------------------------------------------------");
            printf("\nCelular: %s                                  |\tTelefone:
%s", cAux->cvCelular, cAux->cvTelefone);
            printf("\n------------------------------------------------------------
------------------------------------------------------------");
            printf("\nEmail: %s|\tEndereco: %s", cAux->cvEmail, cAux-
>cvEndereco);

printf("\n================================================================================
================================================\n");
            cAux = cAux->cProximo;
        }
        printf("\n\t\t\t\t\t   PRESSIONE QUALQUER TECLA PARA VOLTAR!");
    }
    fflush(stdin);
    getchar();
}

void alteraCliente()
{
      tela("\tALTERAR CLIENTE");
    Cliente *cAux = cClienteInicial;
    char cvCpfCli[201];
    fflush(stdin);
   printf("\nEntre com o CPF do cliente a ser alterado: ");
    strcpy(cvCpfCli,FormatarCPF());
    int iNaoEncontrado = 0;
    while (cAux != NULL)
    {
        if (!strcmp(cAux->cvCpf, cvCpfCli))
        {
            fflush(stdin);
            printf("\nNome: ");
            strcpy(cAux->cvNomeCli,Nome());
            printf("\nCPF: ");
            strcpy(cAux->cvCpf,FormatarCPF());
            printf("\nData de Nascimento: ");
            strcpy(cAux->cvData_nascimento,FormatarData());
            printf("\nEmail: ");
            strcpy(cAux->cvEmail,Nome());
            printf("\nTelefone: ");
            strcpy(cAux->cvTelefone,FormatarTelefone());
            printf("\nCelular: ");
            fflush(stdin);
            strcpy(cAux->cvCelular,FormatarCelular());
            printf("\nEndereco: ");
            strcpy(cAux->cvEndereco,Nome());
            iNaoEncontrado++;
            break;
        }
        cAux = cAux->cProximo;
    }

    if (iNaoEncontrado == 0)
    {
        nullList();
    }
}
```

```
void removeCliente(){
    tela("\tREMOVER CLIENTE");
    Cliente *cAux = cClienteInicial;
    Cliente *cAnterior;
    if(cClienteInicial == NULL){
        nullDelete();
    }else{

        char cvCpfCli[201];
        char cvNomeTemp[200]; //Armazena o nome do item excluido temporariamente
        int iNaoEncontrado = 0;
        fflush(stdin);
        printf("Entre com o CPF do cliente que deseja remover: ");
        strcpy(cvCpfCli,FormatarCPF());
        while(cAux != NULL){
             if(!strcmp(cAux->cvCpf, cvCpfCli)){
                   if(cClienteInicial->cProximo == NULL) cClienteInicial =
NULL;
                   strcpy(cvNomeTemp, cAux->cvNomeCli);
                   if(cAux == cClienteInicial){
                         cAux = cAux->cProximo;
                         free(cClienteInicial);
                               cClienteInicial = NULL;
                               cClienteInicial = cAux;
                               removidoSucesso("Cliente", cvNomeTemp);
                         }else{
                               cAnterior->cProximo = cAux->cProximo;
                               free(cAux);
                               cAux = NULL;
                               removidoSucesso("Cliente", cvNomeTemp);
                               return 0;
                         }
                         iNaoEncontrado++;
                   }
                   cAnterior = cAux;
                   cAux = cAux->cProximo;
             }
             if(iNaoEncontrado == 0) notFound("Cliente");

    }
}

int cadastrarFuncionario(){
    char cEscolha;
    int op;
      if(strcmp(cvAdm, "sim")){
              tela("\t CADASTRO DO ADMINISTRADOR");
             }else{
          tela("\t CADASTRO DE FUNCIONARIO");
      }
          do{
              Funcionario *fNovoFuncionario = (Funcionario *)
malloc(sizeof(Funcionario));
              fNovoFuncionario->funProximo = NULL;

              fflush(stdin);
              printf("\nNome: ");
              strcpy(fNovoFuncionario->cvNomeFunc,Nome());
              if(strcmp(cvAdm, "sim")){
```

```
               strcpy(fNovoFuncionario->cvCargo, "ADMINISTRADOR");
                  fNovoFuncionario->cvCargo[strcspn(fNovoFuncionario->cvCargo,
"\n")] = '\0';
               }else{
                do{
                            printf("\nCargo: ");
                       printf("\n1: ADMINISTRADOR \n2: CAIXA \n3: FINANCEIRO\n");
                       scanf("%d",&op);
                       switch(op){
                            case 1:
                                        strcpy(fNovoFuncionario->cvCargo,
"ADMINISTRADOR");
                                  fNovoFuncionario-
>cvCargo[strcspn(fNovoFuncionario->cvCargo, "\n")] = '\0';
                                  break;
                            case 2:
                                        strcpy(fNovoFuncionario->cvCargo,
"CAIXA");
                                  fNovoFuncionario-
>cvCargo[strcspn(fNovoFuncionario->cvCargo, "\n")] = '\0';
                                  break;
                            case 3:
                                        strcpy(fNovoFuncionario->cvCargo,
"FINANCEIRO");
                                  fNovoFuncionario-
>cvCargo[strcspn(fNovoFuncionario->cvCargo, "\n")] = '\0';
                                  break;
                          default:
                                printf("\nOpcao invalida");
                                fflush(stdin);
                    break;
                        }
               }while((op<1) || (op>3));
         }
                     fflush(stdin);
                printf("\nCPF: ");
                strcpy(fNovoFuncionario->cvCpf,FormatarCPF());
                printf("\nData de Nascimento: ");
                fflush(stdin);
               strcpy(fNovoFuncionario->cvData_nascimento,FormatarData());
               printf("\nEmail: ");
               strcpy(fNovoFuncionario->cvEmail,Nome());
               printf("\nTelefone: ");
               strcpy(fNovoFuncionario->cvTelefone,FormatarTelefone());
               printf("\nCelular: ");
               strcpy(fNovoFuncionario->cvCelular,FormatarCelular());
                printf("\nEndereco: ");
                strcpy(fNovoFuncionario->cvEndereco,Nome());
                CadastroLogin(fNovoFuncionario->cvLogin, fNovoFuncionario-
>cvSenha);

                if(fFuncionarioInicial == NULL) {
                    fFuncionarioInicial = fNovoFuncionario;
                }else{
                    Funcionario *fAux = fFuncionarioInicial;
                    while(fAux->funProximo != NULL){
                        fAux = fAux->funProximo;
                    }
                    fAux->funProximo = fNovoFuncionario;
```

```
              }
             if(strcmp(cvAdm, "sim")){
                    printf("\n\t\tCADASTRO EFETUADO, TESTE SEU LOGIN!!\n\t\t
ENTRANDO...\n");
                    gravarAdm(cvAdm);
             gravarFuncionario();
             Sleep(1600);
              VerificarLogin();
             }else{
              cEscolha = escolhaRegistro();
              tela("\t CADASTRO DE FUNCIONARIO");
      }
          }while(cEscolha != 'n');
          gravarFuncionario();
}

void listarFuncionario(){
      char c[200]="                                                    ";
      int iA;
    tela("\tLISTA DE FUNCIONARIOS");
    if(fFuncionarioInicial == NULL){
        nullList();
    }else{
        Funcionario *fAux = fFuncionarioInicial;
        while(fAux != NULL){
    iA=50 - strlen(fAux->cvNomeFunc);
      strncat(fAux->cvNomeFunc,c,iA);
      iA=49 - strlen(fAux->cvEmail);
      strncat(fAux->cvEmail,c,iA);
                    printf("\n-----------------------------------------------------
-------------------------------------------------------------------");
            printf("Nome: %s|\tCargo: %s", fAux->cvNomeFunc, fAux->cvCargo);
            printf("\n-----------------------------------------------------------
-------------------------------------------------------------");
            printf("\nCPF: %s                                          |\tData de
Nascimento: %s", fAux->cvCpf, fAux->cvData_nascimento);
            printf("\n-----------------------------------------------------------
-------------------------------------------------------------");
            printf("\nCelular: %s                                      |\tTelefone:
%s", fAux->cvCelular, fAux->cvTelefone);
            printf("\n-----------------------------------------------------------
-------------------------------------------------------------");
            printf("\nEmail: %s|\tEndereco: %s", fAux->cvEmail, fAux-
>cvEndereco);

printf("\n=================================================================
=================================================\n");
            fAux = fAux->funProximo;
        }
    printf("\n\t\t\t\t   PRESSIONE QUALQUER TECLA PARA VOLTAR!");
      }
    char cListar;
    cListar = getchar();
    getchar();
}

void alteraFuncionario()
{
      tela("\tALTERAR FUNCIONARIO");
```

```
    Funcionario *fAux = fFuncionarioInicial;
    int iOp;
    char cvCpfFunc[201];
    fflush(stdin);
    printf("\nEntre com o CPF do funcionario a ser alterado: ");
    strcpy(cvCpfFunc,FormatarCPF());
    int iNaoEncontrado = 0;
    while (fAux != NULL)
    {
        if (!strcmp(fAux->cvCpf, cvCpfFunc))
        {
            fflush(stdin);
            printf("\nNome funcionario: ");
            strcpy(fAux->cvNomeFunc,Nome());
            do
            {
                printf("\nCargo: ");
                printf("\n1: ADMINISTRADOR \n2: CAIXA \n3: FINANCEIRO\n");
                scanf("%d", &iOp);
                switch (iOp)
                {
                case 1:
                    strcpy(fAux->cvCargo, "ADMINISTRADOR");
                    fAux->cvCargo[strcspn(fAux->cvCargo, "\n")] = '\0';
                    break;
                case 2:
                    strcpy(fAux->cvCargo, "CAIXA");
                    fAux->cvCargo[strcspn(fAux->cvCargo, "\n")] = '\0';
                    break;
                case 3:
                    strcpy(fAux->cvCargo, "FINANCEIRO");
                    fAux->cvCargo[strcspn(fAux->cvCargo, "\n")] = '\0';
                    break;
                default:
                    printf("\nOpcao invalida");
                    fflush(stdin);
                    break;
                }
            } while ((iOp < 1) || (iOp > 3));
            fflush(stdin);
            printf("\nCPF: ");
            strcpy(fAux->cvCpf,FormatarCPF());
            printf("\nData de Nascimento: ");
             strcpy(fAux->cvData_nascimento,FormatarData());
             printf("\nEmail: ");
             strcpy(fAux->cvEmail,Nome());
             printf("\nTelefone: ");
             strcpy(fAux->cvTelefone,FormatarTelefone());
             fflush(stdin);
             printf("\nCelular: ");
             strcpy(fAux->cvCelular,FormatarCelular());
            printf("\nEndereco: ");
            strcpy(fAux->cvEndereco,Nome());
            iNaoEncontrado++;
            break;
        }
        fAux = fAux->funProximo;
    }
    if (iNaoEncontrado == 0)
```

```
    {
        nullList();
    }
}

void removeFuncionario()
{
    tela("\tREMOVER FUNCIONARIO");
    Funcionario *fAux = fFuncionarioInicial;
    Funcionario *fAnterior;
    if (fFuncionarioInicial == NULL)
    {
        nullDelete();
    }
    else
    {
        char cvCpfFunc[201];
        char cvNomeTemp[200]; // Armazena o nome do item excluido
temporariamente
        int iNaoEncontrado = 0;
        fflush(stdin);
        printf("Entre com o CPF do funcionario que deseja remover: ");
        strcpy(cvCpfFunc,FormatarCPF());

        while (fAux != NULL)
        {
            if (!strcmp(fAux->cvCpf, cvCpfFunc))
            {
                if (fFuncionarioInicial->funProximo == NULL){ //nao pode ser
removido pois eh a chave para o sistema
                                printf("\n\t\t\tESTE FUNCIONARIO NAO PODE SER
REMOVIDO\n");
                                return 0;
                            }

                strcpy(cvNomeTemp, fAux->cvNomeFunc);
                if (fAux == fFuncionarioInicial)
                {
                    fAux = fAux->funProximo;
                    free(fFuncionarioInicial);
                    fFuncionarioInicial = NULL;
                    fFuncionarioInicial = fAux;
                    removidoSucesso("Funcionario", cvNomeTemp);
                }
                else
                {
                    fAnterior->funProximo = fAux->funProximo;
                    free(fAux);
                    fAux = NULL;
                    removidoSucesso("Funcionario", cvNomeTemp);
                    return 0;
                }
                iNaoEncontrado++;
            }
            fAnterior = fAux;
            fAux = fAux->funProximo;
        }
        if (iNaoEncontrado == 0)
            notFound("Funcionario");
```

```
    }
}

int venda(){

	int iFinalizar;
	char cvCpfCliente[201];
	char cvCodProduto[51];
	int iQuantidade = 0;
	float totalItem = 0;
	float totalVenda = 0;
	tela("\t   EFETUAR VENDA");
	//verifica se o cliente esta cadastrado
	tela("\t   EFETUAR VENDA");
	fflush(stdin);
	//laco para item venda
	do{
		if(!iItemVendaInicial == NULL){
			tela("\t   EFETUAR VENDA");
			listaItemVenda(totalVenda);
		}
		//soma total da venda
		totalVenda += addItemVenda();
		printf("Adiconar novo produto? (1 - SIM) / (0 - NAO)");
		scanf("%d", &iFinalizar);
		fflush(stdin);
	}while(iFinalizar != 0);
	if(totalVenda==0.0){
				printf("\n\t\t\t\tVENDA CANCELADA POIS NAO HA
PRODUTOS VENDIDOS\n\n\t\t\t\tPRESSIONE QUALQUER TECLA PARA RETORNAR AO
MENU!!!");
				getchar();
				return 0;
	}
	if (iFinalizar==0){
		tela("\t   EFETUAR VENDA");
	}

	//laco para a venda
	do{

		iFinalizar = addVenda(totalVenda);

	}while(iFinalizar != 1);
	//limpar item venda
	gravarItemVenda();
	iItemVendaInicial = NULL;
	gravarVenda();
	return 0;
}

float addItemVenda(){
	//tela("\t   EFETUAR VENDA");
	//novo item
	ItemVenda *iNovoItem = (ItemVenda*) malloc(sizeof(ItemVenda));
	iNovoItem->iItemProximo = NULL;

	//Variaveis structs auxiliares
	Produto *pAux = pProdutoInicial;
```

```
	//variaveis auxiliares
	char cvCodigoBarra[201];
	int iQuantidade = 0;
	int iNaoEncontrado;
	float totalItem = 0;

	if(pProdutoInicial == NULL){
		printf("Nao ha produtos cadastrados!\n");
		printf("\n\t\t\t\t   PRESSIONE QUALQUER TECLA PARA VOLTAR!");
		fflush(stdin);
	    getchar();
		return 0;
	}

	printf("Entre com o codigo de barras do produto: ");
	strcpy(cvCodigoBarra, Nome());

	while(pAux != NULL){
		if(!strcmp(pAux->cvCodigo, cvCodigoBarra)){
			iNaoEncontrado++;
			strcpy(iNovoItem->cvNomeItem, pAux->cvNome);

	printf("\n**********************************************\nProduto:
%s\t Vl. Unitario: R$
%.2f\n**********************************************\n", pAux->cvNome, pAux-
>fValor);
			printf("Entre com a quantidade de compra: ");
			scanf("%d", &iQuantidade);
			totalItem = pAux->fValor * iQuantidade;
			iNovoItem->fValorItem = totalItem;
			if(pAux->iQtdEstoque < 0){

	printf("**********************************************************************
**************************************************\n");
				printf("\t\t\t\t\tO estoque de %s acabou! Avisar o
Gerente!!\n", pAux->cvNome);

	printf("**********************************************************************
**************************************************\n");
				printf("\n\t\t\t\t   PRESSIONE QUALQUER TECLA PARA
CONTINUAR!");
				fflush(stdin);
				getchar();
			}

			//adicionando na struct de item venda
			iNovoItem->iItemProximo = iItemVendaInicial;
			iItemVendaInicial = iNovoItem;

		}
		pAux = pAux->pProximo;
	}

	if(!iNaoEncontrado){
		printf("\n\t\t\t\tPRODUTO NAO ENCONTRADO!!!\n");
	}

	return totalItem;
```

```
}

int addVenda(float totalVenda){
	int iVerificaCli;
	//variaveis auxiliares
	int iFormaPag;
	float fDinheiroRecebido = 0;
	int iCancelaVenda;

	//colocar na struct de venda
		srand(time(NULL));
		time_t t = time(NULL);
		struct tm tm = *localtime(&t);

		Venda *vNovaVenda = (Venda*) malloc(sizeof(Venda));
		vNovaVenda->vendaProximo = NULL;

		//jogando valores para a nova venda
		vNovaVenda->iCodVenda = rand() % 900;
		vNovaVenda->dTotalCompra = totalVenda;
		vNovaVenda->iDia = tm.tm_mday;
		vNovaVenda->iMes = tm.tm_mon + 1;
		vNovaVenda->iAno = tm.tm_year + 1900;
		listaItemVenda(totalVenda);
		//adm podera alterar
		Cliente *cAux = cClienteInicial;
		iVerificaCli = verificaCliente();
		if(iVerificaCli == 1){
			printf("\nCLIENTE: %s", cvNomeCli);
			if(vNovaVenda->dTotalCompra >= 50){
				vNovaVenda->dTotalCompra = vNovaVenda->dTotalCompra -
(vNovaVenda->dTotalCompra * 0.05);

	printf("\n*********************************************");
				printf("\nValor com desconto: R$ %.2f\n", vNovaVenda-
>dTotalCompra);

	printf("\n*********************************************\n");
			}else if(vNovaVenda->dTotalCompra >= 100){
				vNovaVenda->dTotalCompra = vNovaVenda->dTotalCompra -
(vNovaVenda->dTotalCompra * 0.10);

	printf("\n*********************************************");
				printf("\nValor com desconto: R$ %.2f\n", vNovaVenda-
>dTotalCompra);

	printf("\n*********************************************\n");
			}
		}

		do{
			printf("Forma de pagamento?\n");
			printf("1 - Dinheiro\n");
			printf("2 - Cartao\n");
			scanf("%d", &iFormaPag);
			switch(iFormaPag){
				case 1:
					printf("Dinheiro recebido do cliente: R$ ");
```

```
                                    scanf("%f", &fDinheiroRecebido);
                                    //se nao for igual - ESTOU NEGANDO
                                    if(fDinheiroRecebido > vNovaVenda-
>dTotalCompra){
                                          float fTroco = 0;
                                          fTroco = fDinheiroRecebido - vNovaVenda-
>dTotalCompra;
                                          printf("TROCO: R$ %.2f\n", fTroco);
                                    }
                                    break;
                              case 2: addStructVenda(vNovaVenda);
                        iCancelaVenda = 1;
                        //listaItemVenda(totalVenda);
                        vendaRealizada("cadastrada no sistema");
                        return iCancelaVenda;
                        break;
                              default:
                                    defaultMessage();
                                    break;
                        }
                  }while(iFormaPag != 1);

                  printf("Finalizar venda? (1) | Cancelar Venda (0)\n");
                  scanf("%d", &iCancelaVenda);
                  if(iCancelaVenda == 0){
                        printf("\t\t\t DESEJA REALMENTE CANCELAR A VENDA? (1) - SIM
/ (0) - NAO\n");
                        scanf("%d", &iCancelaVenda);
                        switch(iCancelaVenda){
                              case 0:
                                    //realiza a venda
                                    addStructVenda(vNovaVenda);
                                    //recebe um para parar o loop
                                    iCancelaVenda = 1;
                                    vendaRealizada("realizada");
                                    return iCancelaVenda;
                                    break;
                              case 1:
                                    //limpa a nova venda
                                    free(vNovaVenda);
                                    vNovaVenda = NULL;
                                    //recebe um para parar o loop
                                    iCancelaVenda = 1;
                                    vendaRealizada("cancelada");
                                    return iCancelaVenda;
                                    break;
                        }
                  }else{
                        addStructVenda(vNovaVenda);
                        iCancelaVenda = 1;
                        vendaRealizada("realizada");
                        return iCancelaVenda;
                  }

}

void addStructVenda(Venda *vNovaVenda){
      if(vVendaInicial == NULL){
            vVendaInicial = vNovaVenda;
```

```
	}else{
		Venda *vAux = vVendaInicial;
		while(vAux->vendaProximo != NULL){
			vAux = vAux->vendaProximo;
		}
		vAux->vendaProximo = vNovaVenda;
	}
}

void tiraEstoque(){
	Produto *pAux = pProdutoInicial;
	ItemVenda *iAux = iItemVendaInicial;

	while(pAux != NULL){
		while(iAux != NULL){
			if(!strcmp(pAux->cvNome, iAux->cvNomeItem)){
				printf("Equals\n");
			}
			iAux = iAux->iItemProximo;
		}
		pAux = pAux->pProximo;
	}
}

void listarVenda(){
	tela("\t    LISTA DE VENDAS");
		if(vVendaInicial == NULL){
		nullList();
	}else{
	printf("\nEntre com o dia desejado: ");
	Venda *vAux = vVendaInicial;
	while(vAux != NULL){
		printf("Data: %d/%d/%d\t| Codigo Venda: %d\t|  Total da Venda:
R$%.2f\n", vAux->iDia, vAux->iMes, vAux->iAno,vAux->iCodVenda, vAux-
>dTotalCompra);

	printf("\n==============================================================
============================================================\n");
		vAux = vAux->vendaProximo;
	}
	printf("\n\t\t\t\t\t   PRESSIONE QUALQUER TECLA PARA CONTINUAR!");
}
	fflush(stdin);
	getchar();
}

void vendaRealizada(char mensagem[]){
	printf("**************************************************************
**********************************************\n");
	printf("\t\t\t\t\tVenda %s com sucesso!!!\n", mensagem);
	printf("**************************************************************
**********************************************\n");
	printf("\n\t\t\t\t\t   PRESSIONE QUALQUER TECLA PARA CONTINUAR!");
	fflush(stdin);
	getchar();
}

int pesqVendaData(){
	int iDia, iMes, iAno;
```

```
       int iNaoEncontrado = 0;
       tela("\tLISTA DE VENDAS POR DATA");
       if(vVendaInicial == NULL){
         nullList();
     }else{
       printf("Entre com a data desejada: ");
       scanf("%d/%d/%d", &iDia, &iMes, &iAno);
       Venda *vAux = vVendaInicial;
       while(vAux != NULL){
             if((vAux->iDia == iDia) && (vAux->iMes == iMes) && (vAux->iAno ==
iAno)){
                   printf("\nData: %d/%d/%d\t| Codigo Venda: %d\t| Total da
Venda: R$%.2f\n", vAux->iDia, vAux->iMes, vAux->iAno,vAux->iCodVenda, vAux-
>dTotalCompra);

       printf("\n============================================================
============================================================\n");
             }else{
             notFound("Venda");
             }
             vAux = vAux->vendaProximo;
             printf("\n\t\t\t\t\t   PRESSIONE QUALQUER TECLA PARA VOLTAR!");
             }
             }
             fflush(stdin);
             getchar();
}


void listaItemVenda(float fTotalVenda){
       ItemVenda *iAux = iItemVendaInicial;
       printf("\n\t\t\tProdutos Comprados\n");
       printf("\t\t\t************************************************************
*\n");
       while(iAux != NULL){
             printf("\t\t\tNome Produto: %s\t Valor: %.2f\n", iAux->cvNomeItem,
iAux->fValorItem);
             iAux = iAux->iItemProximo;
       }
       printf("\n\t\t\t\t\t\t\t TOTAL: %.2f\n", fTotalVenda);
       printf("\t\t\t************************************************************
*\n");
}

int verificaCliente(){

       int iNaoEncontrado = 0;
       int iJaCliente= 0;
       char cvCpfCli[201];
       //desconto para clientes
       fflush(stdin);
       printf("\nPossui cadastro? (1 - Sim) / (0 - NAO)");
       scanf("%d", &iJaCliente);
       if(iJaCliente == 1){
             fflush(stdin);
             printf("Entre com o CPF do cliente: ");
             strcpy(cvCpfCli, FormatarCPF());
             Cliente *cAux = cClienteInicial;
             while(cAux != NULL){
```

```
                    if(!strcmp(cAux->cvCpf, cvCpfCli)){
                          iNaoEncontrado=1;
                    strcpy(cvNomeCli, cAux->cvNomeCli);
                    }
                    cAux = cAux->cProximo;
              }

       }
              return iNaoEncontrado;
}

/******* GRAVAR ARQUIVOS *******/
int gravarProduto(){
       FILE *arq;
       arq = fopen("produto.dat", "wb");
       if(arq == NULL){
              printf("Problemas na criacao do arquivo\n");
              return 1;
       }

       Produto *pAux = pProdutoInicial;
       while(pAux != NULL){
              fwrite(pAux, sizeof(Produto),1,arq);
              pAux = pAux->pProximo;
       }
       fclose(arq);
}

int gravarFuncionario(){
       FILE *arq;
       arq = fopen("funcionario.dat", "wb");
       if(arq == NULL){
              printf("\nProblemas na criacao do arquivo");
              return 1;
       }
       Funcionario *fAux = fFuncionarioInicial;
       while(fAux != NULL){
              fwrite(fAux, sizeof(Funcionario),1,arq);
              fAux = fAux->funProximo;
       }
       fclose(arq);
}

int gravarLogin(){
       FILE *arq;
       arq = fopen("login.dat", "wb");
       if(arq == NULL){
              printf("\nProblemas na criacao do arquivo");
              return 1;
       }

       fwrite(iHaveLogin, sizeof(int), 1,arq);
       fclose(arq);
}

int gravarCliente(){
       printf("Entrou aqui\n");
       FILE *arq;
       arq = fopen("cliente.dat", "wb");
```

```
       if(arq == NULL){
              printf("\nProblemas na criacao do arquivo");
              return 1;
       }
       Cliente *fAux = cClienteInicial;
       while(fAux != NULL){
         fwrite(fAux, sizeof(Cliente), 1,arq);
         fAux = fAux->cProximo;
       }
       fclose(arq);
}

int gravarItemVenda(){
       FILE *arq;
       arq = fopen("item_venda.dat", "ab");
       if(arq == NULL){
              printf("\nProblemas na criacao do arquivo");
              return 1;
       }
    ItemVenda *fAux = iItemVendaInicial;
       while(fAux != NULL){
         fwrite(fAux, sizeof(ItemVenda), 1,arq);
         fAux = fAux->iItemProximo;
       }
       fclose(arq);
}

int gravarVenda(){
       FILE *arq;
       arq = fopen("venda.dat", "wb");
       if(arq == NULL){
              printf("\nProblemas na criacao do arquivo");
              return 1;
       }
    Venda *fAux = vVendaInicial;
       while(fAux != NULL){
         fwrite(fAux, sizeof(Venda), 1,arq);
         fAux = fAux->vendaProximo;
       }
    fclose(arq);
}

int gravarFornecedor(){
       FILE *arq;
       arq = fopen("fornecedor.dat", "wb");
       if(arq == NULL){
              printf("\nProblemas na criacao do arquivo");
              return 1;
       }
       Fornecedor *fAux = fFornecedorInicial;
       while(fAux != NULL){
         fwrite(fAux, sizeof(Fornecedor), 1,arq);
         fAux = fAux->fProximo;
       }
       fclose(arq);
}


/******* LER ARQUIVOS *******/
```

```
int lerProduto(){
	Produto pProduto;
	FILE *arq;
	arq = fopen("produto.dat", "rb");
	if(arq == NULL){
		printf("Problemas na criacao do arquivo\n");
		return 1;
	}
	while(!feof(arq)){
		int lido = fread(&pProduto, sizeof(Produto),1,arq);
		if(lido == 1){
			Produto *pAux = (Produto*) malloc(sizeof(Produto));
			strcpy(pAux->cvNome, pProduto.cvNome);
			strcpy(pAux->cvValidade, pProduto.cvValidade);
			pAux->fValor = pProduto.fValor;
			pAux->iQtdEstoque = pProduto.iQtdEstoque;
			pAux->iEstoqSeguranca = pProduto.iEstoqSeguranca;
			strcpy(pAux->cvCodigo, pProduto.cvCodigo);
			pAux->iCodigoFornecedor = pProduto.iCodigoFornecedor;
			pAux->pProximo = pProdutoInicial;
			pProdutoInicial = pAux;
		}
	}
	fclose(arq);

}

int lerFuncionario(){
	Funcionario fun;
	FILE *arq;
	arq = fopen("funcionario.dat", "rb");
	if(arq == NULL){
		printf("\nProblemas na criacao do arquivo!\n");
		return 1;
	}
	while(!feof(arq)){
		int lido = fread(&fun, sizeof(Funcionario),1,arq);
		if(lido == 1){
			Funcionario *newFun = (Funcionario*)
malloc(sizeof(Funcionario));
			//jogando dados do arquivo na struct
			strcpy(newFun->cvNomeFunc, fun.cvNomeFunc);
			strcpy(newFun->cvCargo, fun.cvCargo);
			strcpy(newFun->cvCpf, fun.cvCpf);
			strcpy(newFun->cvData_nascimento, fun.cvData_nascimento);
			strcpy(newFun->cvEmail, fun.cvEmail);
			strcpy(newFun->cvTelefone, fun.cvTelefone);
			strcpy(newFun->cvCelular, fun.cvCelular);
			strcpy(newFun->cvEndereco, fun.cvEndereco);
			strcpy(newFun->cvLogin, fun.cvLogin);
			strcpy(newFun->cvSenha, fun.cvSenha);
			newFun->funProximo = fFuncionarioInicial;
			fFuncionarioInicial = newFun;
		}
	}
	fclose(arq);
}

int lerLogin(){
```

```
	int iLogin;
	FILE *arq;
	arq = fopen("login.dat", "rb");
	if(arq == NULL){
		printf("\nProblemas na criacao do arquivo");
		return 1;
	}

	while(!feof(arq)){
		int lido = fread(iLogin, sizeof(int),1,arq);
		if(lido == 1){
			iHaveLogin = iLogin;
		}
	}
	fclose(arq);
}

int lerCliente(){
	Cliente cCliente;
	FILE *arq;
	arq = fopen("cliente.dat", "rb");
	if(arq == NULL){
		printf("Problemas na abertura do arquivo\n");
		return 1;
	}
	while(!feof(arq)){
		int lido = fread(&cCliente, sizeof(Cliente),1,arq);
		if(lido == 1){
			Cliente *pAux = (Cliente*) malloc(sizeof(Cliente));
			strcpy(pAux->cvNomeCli, cCliente.cvNomeCli);
			strcpy(pAux->cvCpf, cCliente.cvCpf);
			strcpy(pAux->cvData_nascimento, cCliente.cvData_nascimento);
			strcpy(pAux->cvEmail, cCliente.cvEmail);
			strcpy(pAux->cvTelefone, cCliente.cvTelefone);
			strcpy(pAux->cvCelular, cCliente.cvCelular);
			strcpy(pAux->cvEndereco, cCliente.cvEndereco);
			pAux->cProximo = cClienteInicial;
			cClienteInicial = pAux;
		}
	}
	fclose(arq);

}

int lerFornecedor(){
	Fornecedor fFornecedor;
	FILE *arq;
	arq = fopen("fornecedor.dat", "rb");
	if(arq == NULL){
		printf("Problemas na criacao do arquivo\n");
		return 1;
	}
	while(!feof(arq)){
		int lido = fread(&fFornecedor, sizeof(Fornecedor),1,arq);
		if(lido == 1){
			Fornecedor *pAux = (Fornecedor*) malloc(sizeof(Fornecedor));
			strcpy(pAux->cvNome, fFornecedor.cvNome);
			strcpy(pAux->cvNomeFantasia, fFornecedor.cvNomeFantasia);
			strcpy(pAux->cvEndereco, fFornecedor.cvEndereco);
```

```
                    strcpy(pAux->cvCnpj, fFornecedor.cvCnpj);
                    strcpy(pAux->cvEmail, fFornecedor.cvEmail);
                    strcpy(pAux->cvTelefone, fFornecedor.cvTelefone);
                    strcpy(pAux->cvCelular, fFornecedor.cvCelular);
                    pAux->iCodigoProduto = fFornecedor.iCodigoProduto;
                    pAux->fProximo = fFornecedorInicial;
             fFornecedorInicial= pAux;
              }
       }
       fclose(arq);

}


int lerItemVenda(){
       ItemVenda iItemVenda;
       FILE *arq;
       arq = fopen("item_venda.dat", "rb");
       if(arq == NULL){
              printf("Problemas na criacao do arquivo\n");
              return 1;
       }
       while(!feof(arq)){
              int lido = fread(&iItemVenda, sizeof(ItemVenda),1,arq);
              if(lido == 1){
                     ItemVenda *pAux = (ItemVenda*) malloc(sizeof(ItemVenda));
                     strcpy(pAux->cvNomeItem, iItemVenda.cvNomeItem);
                     pAux->fValorItem = iItemVenda.fValorItem;
                     pAux->iItemProximo = iItemVendaInicial;
                     iItemVendaInicial = pAux;
              }
       }
       fclose(arq);

}

int lerVenda(){
       Venda vVenda;
       FILE *arq;
       arq = fopen("venda.dat", "rb");
       if(arq == NULL){
              printf("Problemas na criacao do arquivo\n");
              return 1;
       }
       while(!feof(arq)){
              int lido = fread(&vVenda, sizeof(Venda),1,arq);
              if(lido == 1){
                     printf("Chegou aqui\n");
                     Venda *pAux = (Venda*) malloc(sizeof(Venda));
                     pAux->iDia = vVenda.iDia;
                     pAux->iMes = vVenda.iMes;
                     pAux->iAno = vVenda.iAno;
                     pAux->iCodVenda = vVenda.iCodVenda;
                     pAux->dTotalCompra = vVenda.dTotalCompra;
                     pAux->vendaProximo = vVendaInicial;
                     vVendaInicial = pAux;
              }
       }
       fclose(arq);
```

```
}

void listarEstoque(char nome[]){
    char c[200]="                                                                                                    ";
    char a[200],b[200];
       int iA;
       char d[200]="LISTA DE PRODUTOS - ";
       strcat(d,nome);
    tela(d);
    if(pProdutoInicial == NULL){
         nullList();
    }else{
        Produto *pAux = pProdutoInicial;
        while(pAux != NULL){
        if (!strcmp(pAux->cvSetor, nome)){
    iA=54 - strlen(pAux->cvNome);
       strncat(pAux->cvNome,c,iA);
        sprintf(a, "%d", pAux->iQtdEstoque );
       iA=14 - strlen(a);
       strncat(a,c,iA);
       iA=14 - strlen(pAux->cvCodigo);
       strncat(pAux->cvCodigo,c,iA);
       sprintf(b, "%d", pAux->iEstoqSeguranca );
       iA=13 - strlen(b);
       strncat(b,c,iA);
            printf("Nome: %s| Validade: %s   | Valor: %.2f", pAux->cvNome, pAux-
>cvValidade,pAux->fValor);
            printf("\n----------------------------------------------------------------
----------------------------------------------------------------");
                  printf("\nQuantidade Est: %s| Est.Seguranca: %s| Codigo: %s
| Cod.Fornecedor: %d",a ,b,pAux->cvCodigo, pAux->iCodigoFornecedor);

printf("\n================================================================================
==================================================\n");

    }else{
       }
       pAux = pAux->pProximo;
}
    printf("\n\t\t\t\t   PRESSIONE QUALQUER TECLA PARA VOLTAR!");
}
    fflush(stdin);
    getchar();
    menuEstoque();
}

int gravarAdm(char adm[])
{
  FILE *pont_arq; // cria variï¿½vel ponteiro para o arquivo
        // variï¿½vel do tipo string

  //abrindo o arquivo com tipo de abertura w
  pont_arq = fopen("adm.dat", "wb");

  //testando se o arquivo foi realmente criado
  if(pont_arq == NULL)
  {
  printf("Erro na abertura do arquivo!");
```

```
  return 1;
  }

  strcpy(adm,"sim");

  //usando fprintf para armazenar a string no arquivo
  fprintf(pont_arq, "%s", adm);

  //usando fclose para fechar o arquivo
  fclose(pont_arq);

  printf("Dados gravados com sucesso!");
}

int lerAdm(char adm[])
{
  FILE *pont_arq;

  //abrindo o arquivo_frase em modo "somente leitura"
  pont_arq = fopen("adm.dat", "rb");

  //enquanto nï¿½o for fim de arquivo o looping serï¿½ executado
  //e serï¿½ impresso o texto
  while(fgets(adm, 20, pont_arq) != NULL)

  //fechando o arquivo
  fclose(pont_arq);

}
```

Fonte: elaborado pelos autores.