UNIVERSIDADE FEDERAL DO RIO DE JANEIRO

ESCOLA POLITÉCNICA

DEPARTAMENTO DE ELETRÔNICA E DE COMPUTAÇÃO

Loja Virtual Utilizando Interface WEB/SMS

Autor:

_____

Frederico Andrade de Homobono Balieiro

Orientador:

_____

Prof. Antônio Cláudio Gómez de Sousa

Examinador:

_____

Prof. Aloysio de Castro Pinto Pedroza

Examinador:

_____

Prof. Jose Gabriel Rodriguez Carneiro Gomes

DEL
Agosto de 2008

Resumo

O trabalho desenvolvido com este projeto teve como meta a elaboração de um *sistema web,* ou sistema de informação, que funcionará como uma loja virtual de venda de celulares. Nesse sistema, será implementado funcionalidades não encontradas nas lojas virtuais existentes hoje como a comunicação com o cliente através de *SMS.*

A loja possuirá estoque e qualquer pessoa poderá acessá-la para fazer consulta dos produtos. Para realizar as compras, haverá restrições, o usuário deverá estar cadastrado por questões comerciais. A interface será a mais simples possível de forma que qualquer pessoa sinta-se à vontade para navegar na loja.

O diferencial da loja será o envio de torpedos. Uma vez cadastrado no sistema o cliente poderá receber informações de produtos, promoções e efetuação de compras não só no seu email como também em seu celular. O projeto ajudará na comercialização de celulares.

Palavras-chave

- ***Sistemas de Informação***
- ***Engenharia de Software***
- ***Diagrama de Fluxo de Dados***
- ***Web***
- ***SMS***

Glossário

GSM – Global System for Mobile Communications.

PHP – Php Hypertext Preprocessor.

SMS – Short Message Service.

VAS – Value Added Service.

HTTP – HyperText Transfer Protocol.

Índice

1 Introdução

1.1 Assunto

Sistema de Informação é a expressão utilizada para descrever um sistema automatizado (que pode ser denominado como Sistema de Informação Computadorizado), ou mesmo manual, que abrange pessoas, máquinas, e/ou métodos organizados para coletar, processar, transmitir e disseminar dados que representam informação para o usuário.

A área de conhecimento *Sistemas de Informação* é uma área multi ou trans-disciplinar, devido às inter-relações com outras áreas de conhecimento, tais como Ciência da Computação, Administração, Gestão da Informação, Economia, Sociologia, Direito, Engenharia de Produção, Ciência da Informação e outras.

As concepções mais modernas de *Sistemas de Informação* contemplam também os Sistemas de telecomunicações e/ou equipamentos relacionados; sistemas ou subsistemas interconectados que utilizam equipamentos na aquisição, armazenamento, manipulação, gestão, movimento, no controle, na exposição, na troca, no intercâmbio, na transmissão, ou na recepção da voz e/ou dos dados, e inclui o software e hardware utilizados.

Um Sistema de Informações pode ser então definido como todo sistema usado para prover informação (incluindo o seu processamento), qualquer que seja o uso feito dessa informação. Este tipo de sistema possui vários elementos inter-relacionados que coletam (entrada), manipulam e armazenam (processo), disseminam (saída) os dados e informações e fornecem um mecanismo de feedback.

1.2 Motivação

O fator motivador que levou à criação deste projeto é o desejo de adquirir conhecimentos na área de desenvolvimento e implantação de sistemas, pois é um setor da tecnologia que envolve tantos desafios de negócios quanto técnicos. Outro fator motivador foi a realização de um estágio em uma multinacional de telecomunicações que permitiu o contato com equipamentos deste ramo e estudo de programação utilizando sockets. A experiência adquirida no estágio permitiu a criação de um módulo *GSM* que seria responsável por envio de torpedos através de um modem *GSM* e do envio de informações, relativas ao sistema, através de emails utilizando sockets.

O mercado de Internet na América Latina está crescendo mais rápido do que em qualquer outra região. De 2002 para 2005, tivemos um crescimento de 32 milhões para 58 milhões de usuários. A América Latina tem 2,5 milhões de domínios registrados na Internet, sendo que 32% deles são domínios .com e .net. Com mais de 22 milhões de usuários de Internet, aproximadamente 12% da população, o Brasil é um dos maiores mercados de Internet da região e apresenta a maior penetração de banda larga.

Além disso, a *Internet* tem se mostrado cada vez mais como uma importante ferramenta para o terceiro setor. Além de ser um espaço de comunicação e de divulgação onde qualquer organização pode construir sua página, a *web* tem sido um ótimo meio para a prática da economia solidária e do comércio justo.

1.3 Objetivos

O objetivo desse projeto é o desenvolvimento de um sistema de venda de celulares. Neste sistema o usuário poderá fazer diversos tipos de busca baseada em banco de dados. Também haverá um sistema de gerência administrativa, tal como controle de estoque.

Além disto, o cliente poderá ser notificado através de torpedo *SMS*, sobre novos produtos da loja, compras realizadas e notificado sobre novas promoções.

O aplicativo a ser desenvolvido é um sistema independente. Uma vez que suas atribuições permitem, além da venda de celulares, a administração da loja e o controle de estoque, não é necessário integrá-lo a outros sistemas.

Os seguintes atributos de qualidade podem ser atribuídos ao projeto:

- Amigabilidade – Uma Loja online, para atrair clientes, precisa ter um ambiente agradável para o usuário, que o faça se sentir bem navegando. Além disso, é necessário que o ambiente seja de fácil utilização, uma vez que a grande maioria dos clientes não possui grandes conhecimentos técnicos.
- Segurança – Uma vez que dados pessoais de cliente serão armazenados em um banco de dados, é necessário que exista um bom sistema de segurança.
- Manutenível - O sistema que estamos projetando tem essa característica devido a estar sendo bem especificado, e a codificação será feita com muitos comentários, o que facilitará a correção de erros e a expansão. Essa última tem uma maior importância devido ao fato de que sempre é necessário atribuir novas funções para o melhor funcionamento de uma loja via *web*.

2 Planejamento

2.1 Análise do Problema

A partir da idéia inicial e da definição dos objetivos do projeto, iniciou-se o processo de análise do problema. Surgiu, então, a necessidade da implementação das mais variadas técnicas que abrangessem as funcionalidades que o sistema demandava.

Como tais técnicas abrangem áreas bem variadas da Engenharia Eletrônica e de Computação, foi decidido que as suas apresentações iam ser divididas por segmentos de atuação, como mostrados a seguir:

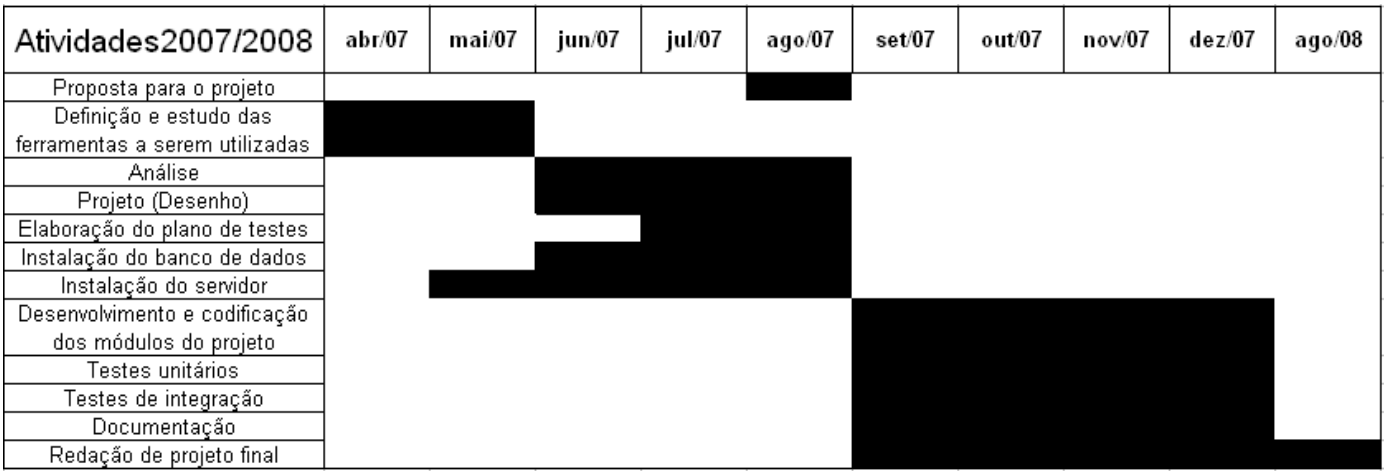
- Desenvolvimento *web*:
  - Desenvolvimento de um site com interface amigável, de forma que usuários não ambientados com o sistema possam navegar facilmente.
- Linguagem de Programação:
  - Codificação do sistema.
  - Possibilidade de implementação de novas funcionalidades, alterando-se pequenos trechos do código fonte.
- Banco de Dados:
  - Utilização de um sistema de banco de dados para o controle dos usuários cadastrados no sistema.
- Engenharia de *Software*:
  - Planejamento, análise e projeto do sistema.
- Módulo *GSM*
  - Instalação de um modem *GSM*
  - Implementação de um *script* para envio de *SMS*.

Tendo sido definidos os requisitos que o sistema deve atender, o capítulo seguinte vai tratar da pesquisa de informações para a implementação de cada área apresentada acima.

2.2 Cronograma

Para garantir a qualidade, o andamento satisfatório e a execução das tarefas de forma ordenada, foi elaborado um cronograma que englobou todas as atividades referentes ao projeto.

Este é apresentado a seguir e conta com a data de início, a data limite para o fim de cada atividade e uma breve descrição da atividade.



As atividades de relacionadas a documentação foram realizadas até agosto de 2008.

3 Coleta de Informações

Neste capítulo será mostrado o estudo de cada divisão do projeto, compreendendo o estudo de técnicas, métodos, ambientes e tecnologias a serem utilizados.

3.1 Ambiente

Como o presente projeto é um sistema cuja interface com o usuário é baseado em navegador, é essencial apresentação das ferramentas a serem utilizadas.

3.1.1 Apache

O servidor Apache (ou Servidor HTTP Apache, em inglês: Apache HTTP Server, ou simples: Apache) é o mais bem sucedido servidor *web* livre. Foi criado em 1995 por Rob McCool, então funcionário do NCSA (National Center for Supercomputing Applications). Numa pesquisa realizada em dezembro de 2005, foi constatado que a utilização do Apache supera 60% nos servidores ativos no mundo.

O servidor é compatível com o protocolo HTTP versão 1.1. Suas funcionalidades são mantidas através de uma estrutura de módulos, podendo inclusive o usuário escrever seus próprios módulos — utilizando a API do software. É disponibilizado em versões para os sistemas Windows, Novell Netware, OS/2 e sistemas no padrão POSIX (Unix, Linux, FreeBSD, etc).

Para garantir segurança nas transações HTTP, o servidor dispõe de um módulo chamado mod_ssl, para atender requisições utilizando o protocolo HTTPS. Este protocolo utiliza uma camada SSL para criptografar todos os dados transferidos entre o cliente e o servidor, provendo maior grau de segurança, confidencialidade e confiabilidade dos dados. A camada SSL é compatível com certificados X.509, que são os certificados digitais fornecidos e assinados por grandes entidades certificadoras no mundo.

(*) Para maiores informações, consulte a bibliografia.

3.2 *Linguagem de Programação*

A linguagem de programação escolhida para o desenvolvimento da parte *web* do projeto foi PHP. Esta linguagem é muito utilizada para gerar conteúdo dinâmico na World Wide Web, além de ser uma linguagem de fácil aprendizagem e de utilização para pequenos *scripts* dinâmicos simples.

A linguagem PHP é uma linguagem de programação de domínio específico, ou seja, seu escopo se estende a um campo de atuação que é o desenvolvimento *web*. Seu propósito principal é de implementar soluções *web* velozes, simples e eficientes.

Características:

- Multiplataforma
- Estruturada e orientada a objeto
- Portável - independência de plataforma - escreva uma vez, rode em qualquer lugar;
- Sintaxe similar à linguagem C/C++

Construir uma página dinâmica baseada em bases de dados é simples com PHP, este provê suporte a um grande número de bases de dados: Oracle, Sybase, PostgreSQL, InterBase, MySQL, SQLite, MSSQL, Firebird, etc. PHP tem suporte aos protocolos: IMAP, SNMP, NNTP, POP3, HTTP, LDAP, XML-RPC, SOAP. É possível abrir sockets e interagir com outros protocolos. E as bibliotecas de terceiros expandem ainda mais estas funcionalidades.

Figura 1 – Diagrama PHP

3.3 Banco de Dados

Para o controle de acesso dos usuários cadastrados é necessária a utilização de um sistema gerenciador de banco de dados (SGBD). Como sua utilização não seria muito complexa e desejando um sistema de utilização livre, de interface amigável e de estabilidade garantida foi escolhido o MySQL (versão 5.0.24a).

O sucesso do MySQL deve-se em grande medida à fácil integração com o PHP, devido a facilidade de instalação no servidor de hospedagem, ambos são gratuitos e são compatíveis. É utilizado largamente em hospedagem de sites da Internet oferecidos atualmente. Empresas como Yahoo! Finance, MP3.com, Motorola, NASA, Silicon Graphics e Texas Instruments usam o MySQL em suas aplicações.

Para a integração do banco de dados com o sistema, bastou a criação de variáveis para receber a conexão e mais outras para usar os comandos SQL usados na filtragem de dados e na impressão dos mesmos.

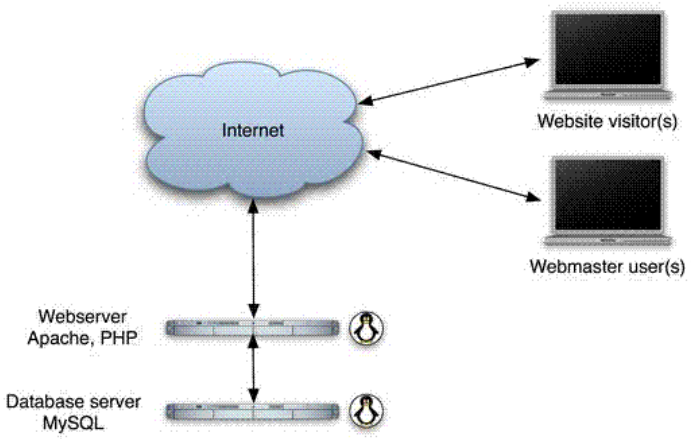


Figura 2 - Conexão da aplicação com o banco de dados.

3.4 Engenharia de Software

Como este projeto trata do desenvolvimento de um software, esta etapa é de extrema importância, pois é através dela que os próximos passos do projeto serão determinados. O trabalho de Engenharia de Software consiste na utilização sistemática, disciplinada e quantificada de um conjunto de técnicas para o desenvolvimento, operação e manutenção de um sistema.

Devido a crescente complexidade dos sistemas elaborados, desenvolveram-se estratégias de desenvolvimento que englobam a metodologia e o processo.

Será utilizada a metodologia estruturada com a finalidade de retratar o fluxo e o conteúdo das informações utilizadas pelo sistema, dividir o sistema em partições funcionais e comportamentais e descrever a essência daquilo que será construído.

O modelo de processo que utilizaremos neste projeto é o chamado Modelo Linear Seqüencial, também conhecido como ciclo de vida clássico.

Suas etapas serão mostradas na figura abaixo:

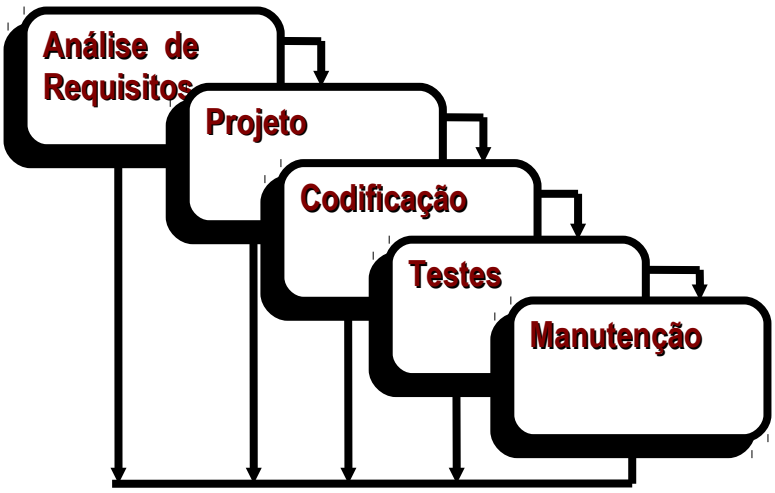


Figura 3 – Modelo Linear de desenvolvimento de software

- Análise: O sistema reconhecerá três tipos de usuários: o administrador, o cliente e o visitante. O administrador poderá incluir, alterar e remover produtos enquanto o usuário deverá ser capaz de buscar e realizar compras dos produtos disponíveis em estoque. O visitante terá permissão para consultar os produtos da loja. O sistema não deverá revelar aos operadores nehuma informação pessoal sobre os clientes. Para esta

etapa serão utilizados Diagrama de Fluxo de Dados, Dicionário de Dados, Diagrama entidade relacionamento e Diagrama de Estados.

- Projeto: Esta etapa é responsável pela transformação dos requerimentos do sistema na sua arquitetura. Serão utilizados padrões de projeto de software que são necessários para uma divisão bem definida do sistema. Haverá um servidor, onde ficarão armazenados os dados, conectado ao modem *GSM*. Os dados ficarão armazenados no servidor e o tráfego dos mesmos será realizado pela Internet.

- Codificação: O Projeto deve ser transformado em uma linguagem de programação a ser interpretada.

- Teste: Depois que o código foi gerado, inicia-se a fase de teste, para assegurar a confiabilidade do sistema. Testes unitários, de integração e de validação.

- Manutenção: O software poderá sofrer mudanças depois que for entregue ao cliente.

4 Modelagem da Solução

Conforme foi dito no item referente à Engenharia de Software a modelagem será feita nos próximos capítulos seguindo as etapas da Análise e Projeto.

4.1 Análise

Para o projeto foi escolhida a análise estruturada que é uma atividade de construção de modelos. Utiliza uma notação que é própria ao método de análise estruturada com a finalidade de retratar o fluxo e o conteúdo das informações utilizadas pelo sistema, dividir o sistema em partições funcionais e comportamentais e descrever a essência daquilo que será construído.

4.1.1 Lista de Eventos

A lista de eventos é uma lista narrativa dos "estímulos" que ocorrem no mundo exterior, e os quais o sistema deve responder. A seguir segue a listagem dos eventos do projeto.

- Fazer o cadastro no Sistema
- Fazer logon no Sistema
- Recuperar senha do Usuário
- Fazer logoff no Sistema
- Remover usuário do Sistema
- Editar dados pessoais
- Alterar senhas
- Visualizar produtos (celulares)
- Pesquisar produtos (celulares)
- Visualizar marcas
- Pesquisar marcas
- Excluir do Sistema
- Comprar produtos (celulares)
- Colocar produtos no carrinho
- Colocar produtos nos favoritos
- Agendar envio de alertas
- Cadastrar produtos (celulares)
- Editar produtos (celulares)
- Remover produtos (celulares)
- Atualizar estoque
- Cadastrar marcas

- Editar marcas
- Remover marcas

A arquitetura do sistema será a seguinte:

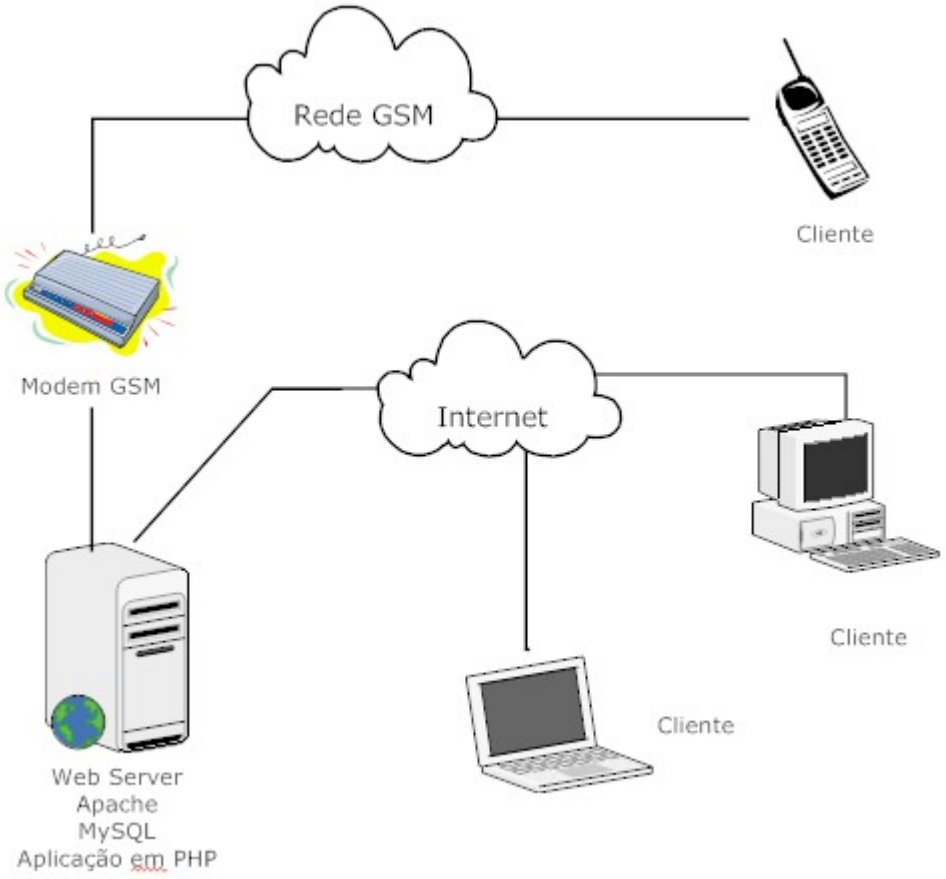


Figura 4 – Arquitetura do sistema

Como se pode notar os servidores ficaram armazenados em uma mesma máquina. O meio por onde os dados trafegarão será a Internet. Como dito anteriormente, o servidor será Apache e o banco de dados será o MySQL.

4.1.2 Diagrama de Entidade Relacionamento

O sistema usa o modelo conceitual de dados que está representado no seguinte DER.

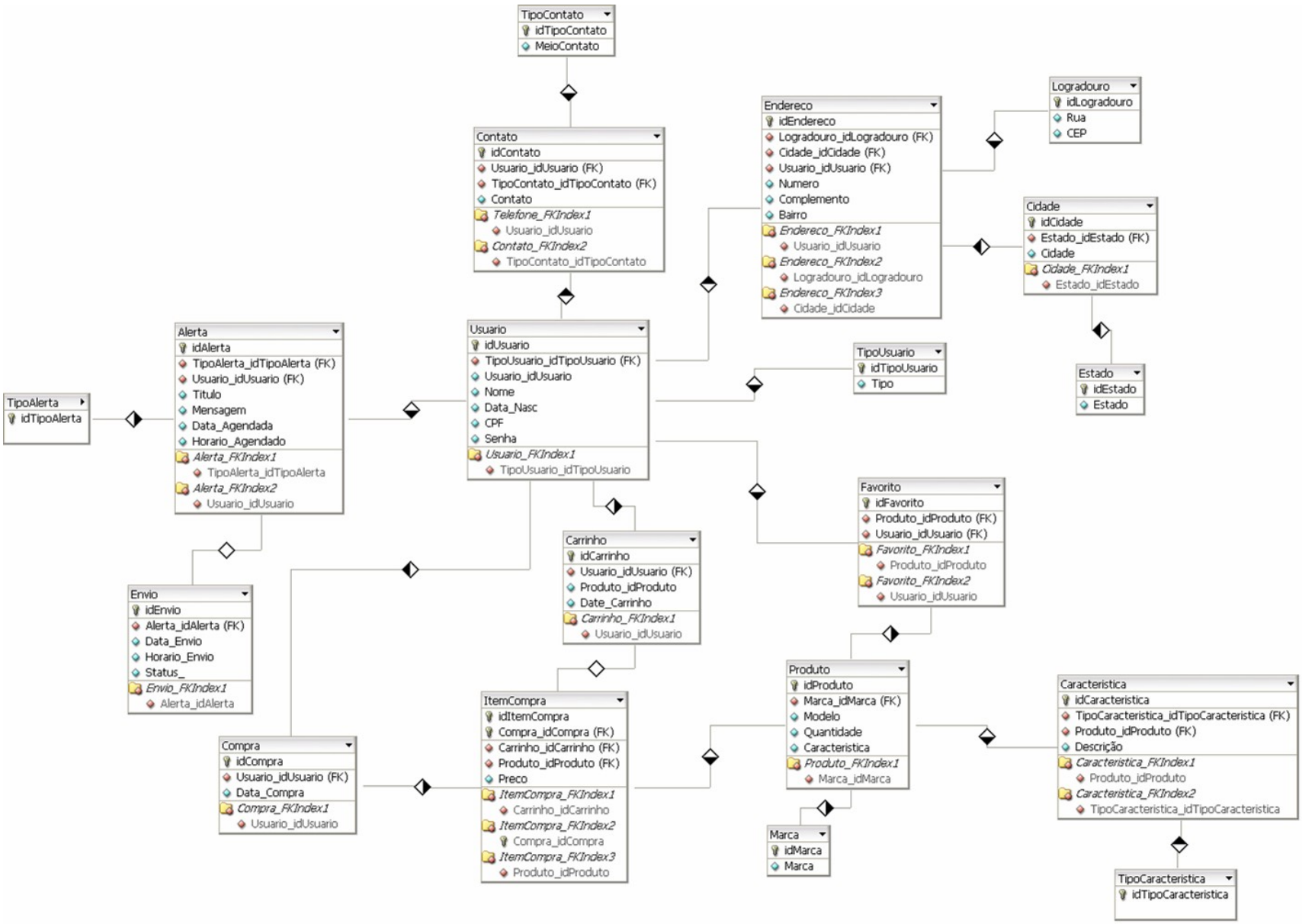

TipoContato
idTipoContato
MeioContato
Contato
idContato
Usuario_idUsuario (FK)
TipoContato_idTipoContato (FK)
Contato
Telefone_FKIndex1
Usuario_idUsuario
Contato_FKIndex2
TipoContato_idTipoContato
Endereco
idEndereco
Logradouro_idLogradouro (FK)
Cidade_idCidade (FK)
Usuario_idUsuario (FK)
Numero
Complemento
Bairro
Endereco_FKIndex1
Usuario_idUsuario
Endereco_FKIndex2
Logradouro_idLogradouro
Endereco_FKIndex3
Cidade_idCidade
Logradouro
idLogradouro
Rua
CEP
Cidade
idCidade
Estado_idEstado (FK)
Cidade
Cidade_FKIndex1
Estado_idEstado
Estado
idEstado
Estado
Alerta
idAlerta
TipoAlerta_idTipoAlerta (FK)
Usuario_idUsuario (FK)
Titulo
Mensagem
Data_Agendada
Horario_Agendado
Alerta_FKIndex1
TipoAlerta_idTipoAlerta
Alerta_FKIndex2
Usuario_idUsuario
TipoAlerta
idTipoAlerta
Usuario
idUsuario
TipoUsuario_idTipoUsuario (FK)
Usuario_idUsuario
Nome
Data_Nasc
CPF
Senha
Usuario_FKIndex1
TipoUsuario_idTipoUsuario
TipoUsuario
idTipoUsuario
Tipo
Favorito
idFavorito
Produto_idProduto (FK)
Usuario_idUsuario (FK)
Favorito_FKIndex1
Produto_idProduto
Favorito_FKIndex2
Usuario_idUsuario
Carrinho
idCarrinho
Usuario_idUsuario (FK)
Produto_idProduto
Date_Carrinho
Carrinho_FKIndex1
Usuario_idUsuario
Envio
idEnvio
Alerta_idAlerta (FK)
Data_Envio
Horario_Envio
Status_
Envio_FKIndex1
Alerta_idAlerta
Compra
idCompra
Usuario_idUsuario (FK)
Data_Compra
Compra_FKIndex1
Usuario_idUsuario
ItemCompra
idItemCompra
Compra_idCompra (FK)
Carrinho_idCarrinho (FK)
Produto_idProduto (FK)
Preco
ItemCompra_FKIndex1
Carrinho_idCarrinho
ItemCompra_FKIndex2
Compra_idCompra
ItemCompra_FKIndex3
Produto_idProduto
Produto
idProduto
Marca_idMarca (FK)
Modelo
Quantidade
Caracteristica
Produto_FKIndex1
Marca_idMarca
Marca
idMarca
Marca
Caracteristica
idCaracteristica
TipoCaracteristica_idTipoCaracteristica (FK)
Produto_idProduto (FK)
Descrição
Caracteristica_FKIndex1
Produto_idProduto
Caracteristica_FKIndex2
TipoCaracteristica_idTipoCaracteristica
TipoCaracteristica
idTipoCaracteristica

Figura 5 – Diagrama Entidade Relacionamento

Neste modelo as chaves estrangeiras aparecem para garantir a integridade dos dados.

Para o DER temos o seguinte dicionário de dados:

- **Produto –** Entidade que define dados relativos ao produto.
    - **idProduto –** Código que identifica o produto.
    - **Marca_idMarca –** Código da marca do produto.
    - **Modelo –** Modelo do produto.
    - **Quantidade –** Quantidade do produto no estoque.

- **Característica –** Guarda informações relativas as características do produto..
    - **idCaracteristica –** Código que identifica a característica do produto.
    - **Característica –** Descrição das características do produto.

- **TipoCaracterística –** Guarda informações relativas ao tipo de características do produto..
    - **idTipoCaracteristica –** Código que identifica o tipo de característica do produto.
    - **TipoCaracterística –** Tipo de características do produto.

- **Marca -** Define os vários tipos de marcas do produto.
    - **idMarca –** Código que identifica a marca do produto.
    - **Marca –** Nome da marca do produto.
- **Favorito –** Lista de todos os produtos favoritos do usuário.
    - **idFavorito -** Código que identifica o favorito.
    - **Produto_idProduto -** Código que identifica o produto favorito.
    - **Usuário_idUsuario -** Código que identifica o usuário.

- **Carrinho –** Entidade que define as compras do usuário.
    - **idCarrinho -** Código que identifica o carrinho.
    - **Data_Carrinho –** Data de compra dos produtos.

- **ItemCompra –** Contém a informação de todos as compras realizadas..
    - **idItemCompra -** Código que identifica o item de compra.
    - **Preço –** Preço da compra.

- **Compra –** Guarda informações relativas a compra**.**
    - **idCompra -** Código que identifica uma compra do usuário.
    - **Data_Compra –** Data em que a compra foi realizada**.**

- **Usuario –** Guarda informações relativas ao usuário.
    - **Id_Usuario -** Código que identifica o usuário.
    - **Nome -** Nome do usuário.
    - **Data_Nasc –** Data de nascimento do usuário**.**
    - **CPF –** CPF do usuário.
    - **Senha** – Senha do usuário**.**

- **TipoUsuario –** Guarda informação do tipo de usuário (Cliente ou administrador).
    - **idTipoUsuario -** Código que identifica o tipo de usuário.
    - **Tipo –** Guarda a informação sobre o tipo de usuário (Cliente ou administrador).

- **Contato –** Guarda informações relativas ao contato do usuário.
    - **idContato -** Código que identifica o contato do usuário.
    - **Contato –** Guarda o contato do usuário.

- **TipoContato** - Guarda informações relativas ao tipo de contato do usuário.
    - **idTipoContato -** Código que identifica o tipo de contato.
    - **MeioContato** – Guarda o meio de contato do usuário. Pode ser por exemplo e-mail, telefone residencial, celular, etc.

- **Alerta -** Lista as formas possíveis de alerta (e-mail ou *sms*).
    - **idAlerta -** Código que identifica o alerta.
    - **Título –** Título do alerta que será encaminhado.
    - **Mensagem –** Mensagem que será enviada ao usuário**.**
    - **Data_Agendada –** Data em que o alerta será enviado**.**
    - **Horário_Agendado –** Horário em que o alerta será enviado.

- **TipoAlerta -** Apresenta o tipo de alerta.
    - **idTipoAlerta -** Código que identifica o tipo de alerta.
    - **Tipo –** Nome do tipo de alerta (e-mail ou *sms*).

- **Endereço –** Guarda informações relativas ao endereço do usuário.
    - **idEndereco -** Código que identifica o endereço.
    - **Numero –** Número da residência do usuário.
    - **Complemento –** Complemento da residência**.**
    - **Bairro –** Bairro da residência**.**
- **Logradouro –** Contém as informações relativas ao endereço do usuário**.**
    - **idLogradouro -** Código que identifica o logradouro.
    - **Rua -** Nome do endereço do usuário.
    - **CEP –** CEP do endereço**.**

- **Cidade –** Lista as cidades que podem ser escolhidas no sistema.
    - **idCidade -** Código que identifica a cidade.
    - **Cidade –** Nome da cidade.

- **Estado –** Lista os estados que podem ser escolhidos no sistema.
    - **idEstado -** Código que identifica o estado.
    - **Estado -** Nome do estado.

- **Envio –** Contém a informação do status de envio do e-mail ou *SMS*. Pode ser enviado ou não enviado
    - **Data_Envio –** Data de envio da mensagem.
    - **Horário_Envio –** Horário de envio da mensagem.

4.1.3 Diagrama de Fluxo de dados

O DFD ou Diagrama de Fluxos de Dados é uma ferramenta para a modelagem de sistemas. Ela fornece apenas uma visão do sistema, a visão estruturada das funções, ou seja, o fluxo dos dados.

Segue a descrição dos DFD's

4.1.3.1 DFDs para visitante

- Ver produto

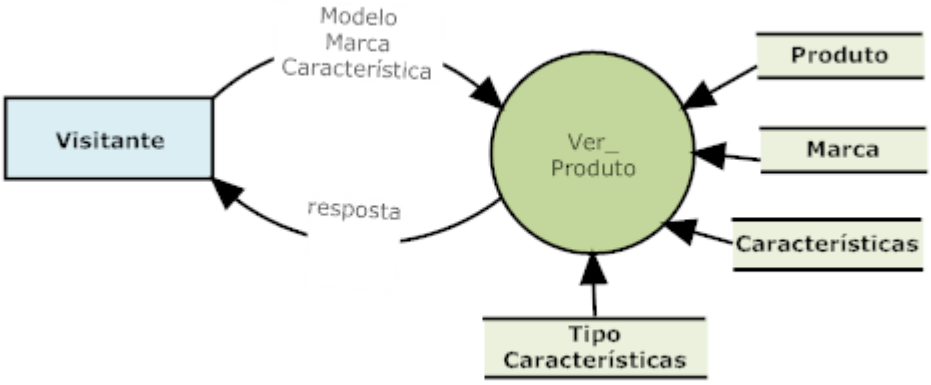


Figura 6 – DFD Ver Produto

Descrição do processo primitivo:

- *Ver_Produto*
  - Objetivos: Mostrar informações sobre um determinado produto para o usuário que não for cadastrado na loja. O usuário não cadastrado é tratado como visitante pelo sistema.

Recebe o código do produto e busca as informações associadas ao produto nos depósitos de dados "Produto", "Modelo", "Marca" e "Características". Os dados do produto são formatados e a página com as informações é enviada para o cliente.

- Cadastrar cliente

Uma vez que o usuário não esteja cadastrado, ele é tratado como visitante pelo sistema. Desta maneira temos o seguinte DFD para o usuário não cadastrado no sistema.

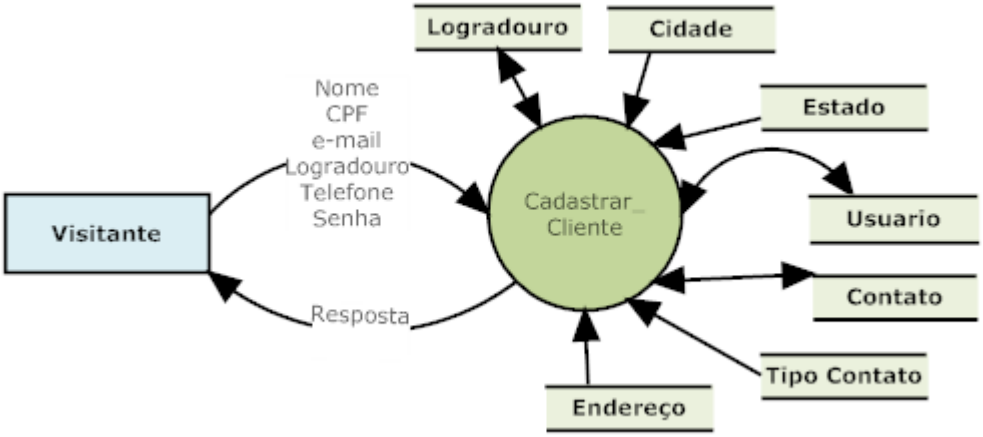


Figura 7 – DFD Cadastrar Cliente

Por se tratar de uma função muito complexa, foi feita a explosão do processo da seguinte forma:

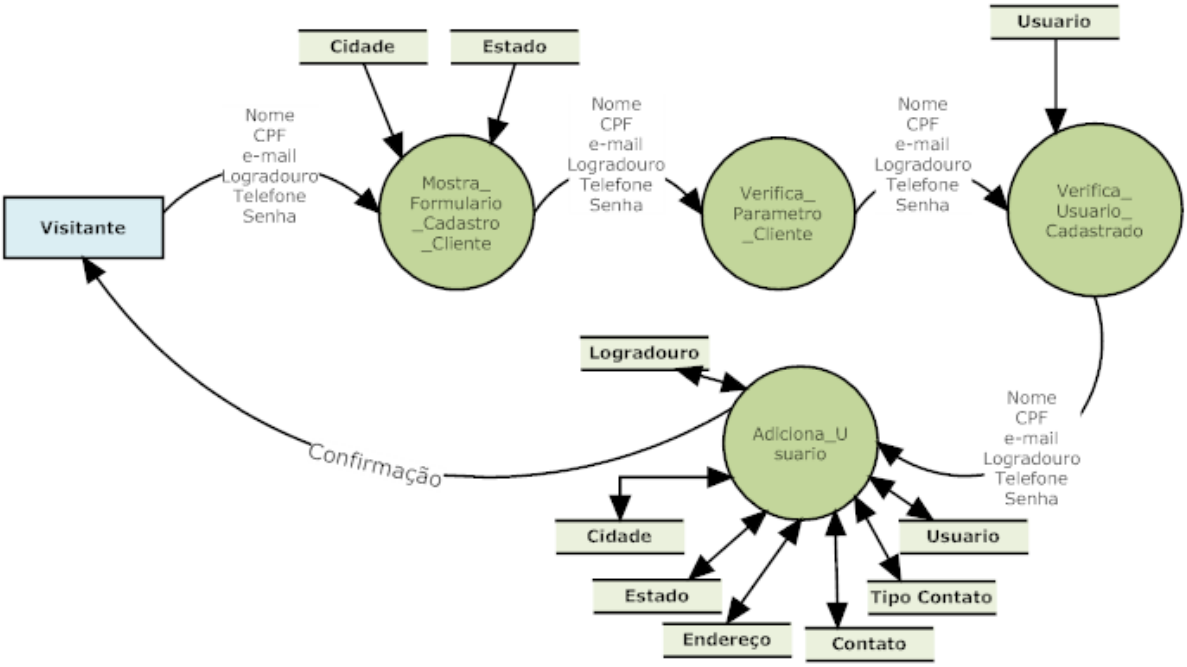


Figura 8 – DFD explodido do Cadastrar Cliente

Especificação dos processos primitivos:

- *Mostra_Formulario_Cadastro_Cliente*

  - Objetivos: Mostrar ao visitante um formulário para que ele possa preencher com os dados para cadastro.

  A função consulta o depósito de dados cidade e estado, para gerar a lista dos mesmos, permitindo que o cliente escolha uma entre as opções. Quando o cliente terminar de preencher os campos, envia os dados para a Verifica_Parametros_Cliente.

- *Verifica_Parametros_Cliente*

  - Objetivos: Verificar se os parâmetros celular, telefone, CPF, e-mail e nome são válidos e se existe algum campo vazio.

Os parâmetros passados pela função anterior são recebidos.
Se existir algum campo vazio, retornar mensagem de erro.
Se os campos celular e telefone não forem preenchidos oito dígitos apenas numéricos, retornar mensagem de erro.
Se o campo CPF contiver um valor inválido, retornar mensagem de erro.
Se o e-mail não tiver um formato adequado, retornar mensagem de erro.
Se o nome forem menor que quatro caracteres, retornar erro.
Caso nenhuma das opções acima aconteça, repassar os dados para a próxima função.

- *Verifica_Usuario_Cadastrado*

  - Objetivos: Verificar se o usuário já possui registro na base de dados.

  Recebe os parâmetros da função anterior, e, com o nome do usuário, verifica se ele já está cadastrado.
  Se não estiver, repassa os parâmetros a frente.
  Se estiver, retorna mensagem de erro.

- *Adiciona_Usuário*

  - Objetivos: Cadastrar o novo usuário na base de dados.

  Recebe os parâmetros, e os cadastra no depósito de dados cliente. Em seguida, envia uma mensagem de sucesso ao cliente.

4.1.3.2 DFDs para cliente

- Login de cliente

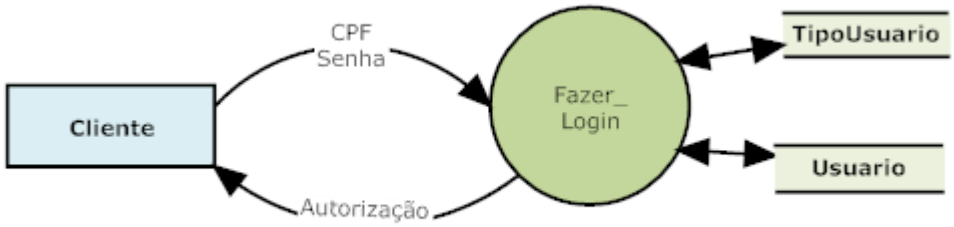


Figura 9 – DFD Login do cliente

Descrição do processo primitivo:

- *Fazer_Login:*

  - Objetivos: Autenticar o cliente, permitindo que ele faça uma compra.

A função recebe o CPF e a senha criptografada do cliente e consulta "Usuário" para saber se elas são compatíveis.

Se o login não existir na base de dados, retorna mensagem de usuário não cadastrado.

Se a senha for correta para aquele cliente, é enviada uma autorização para ele. Senão, é enviada uma mensagem de acesso não autorizado.

- Logoff de cliente

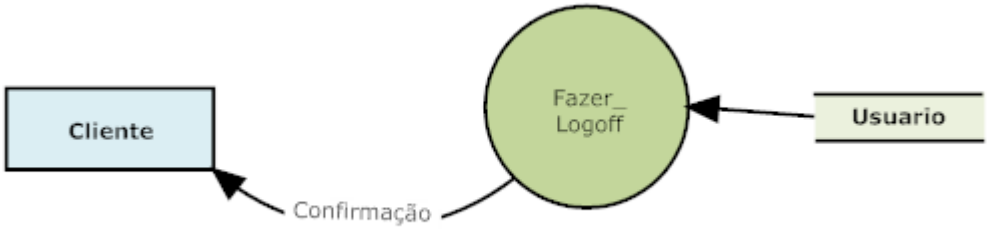


Figura 10 – DFD Logoff

Descrição do processo primitivo:

- *Logoff_Cliente*
  - Objetivos: Desconectar o cliente do site.

Quando for recebido um pedido de logoff, a função irá destruir a autorização do cliente, e enviará uma mensagem de operação realizada com sucesso.

- Editar cadastro do cliente

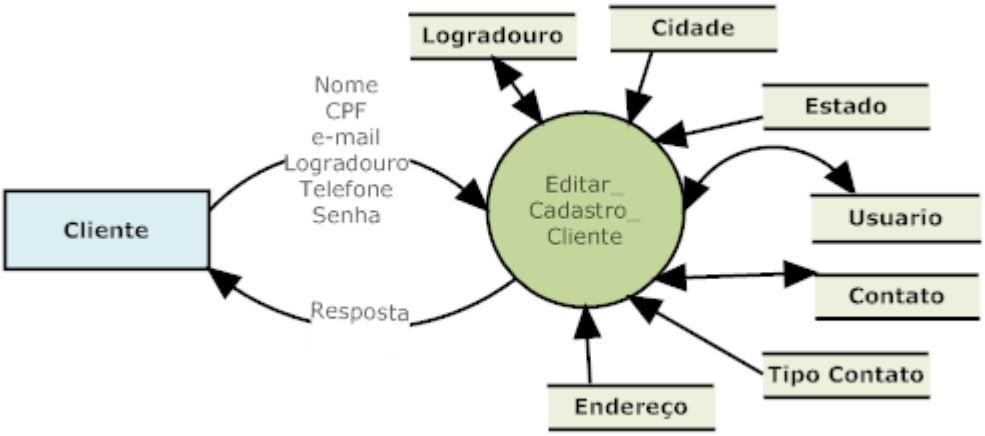


Figura 11 – DFD Editar cadastro do cliente

Por se tratar de uma função muito complexa, foi feita a explosão do processo da seguinte forma:

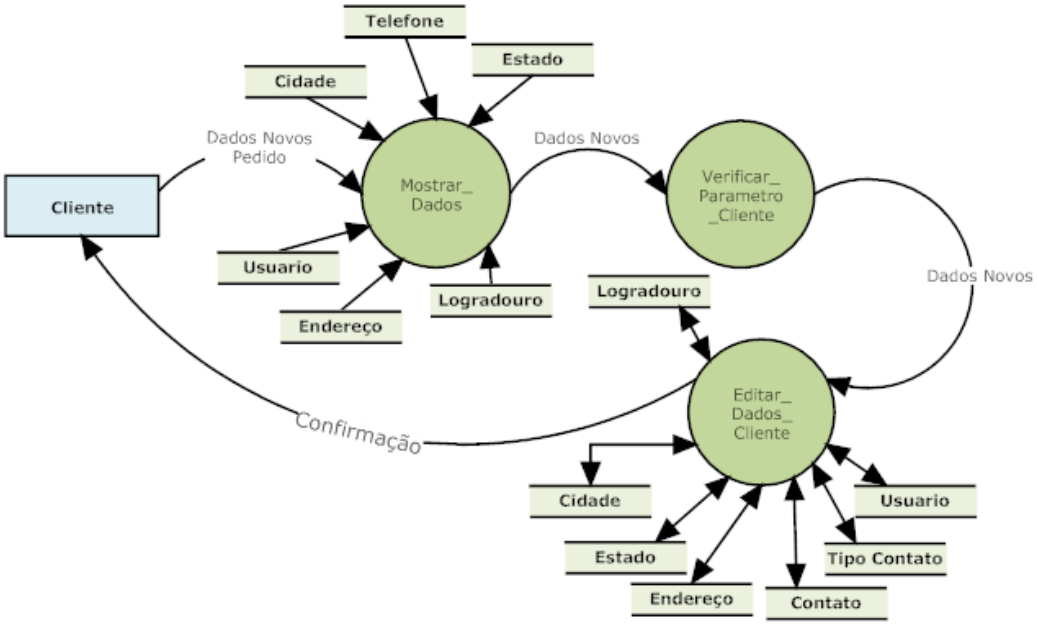


Figura 12 – DFD explodido do Editar cadastro do cliente

Especificação dos processos primitivos:

- *Mostra_Dados*

    - Objetivos: Mostar os dados já cadastrados e receber os dados novos.

    A função recebe um pedido para editar os dados do cliente, e então consulta a base de dados para obter todas as informações do mesmo. Em seguida, o cliente edita os dados que estão sendo exibidos, e ao fim dessa atividade, a função repassa esses parâmetros para a Verifica_Parametros_Cliente.

- *Verifica_Parametros_Cliente*: Já definida em **Cadastrar_Cliente**.

- *Edita_Dados_Cliente:*

    - Objetivos: Editar a base de dados com os dados recebidos.

    Essa função recebe os novos dados do cliente, e os insere na base de dados. Em seguida, envia uma mensagem de confirmação ao cliente.

- Esqueceu Senha

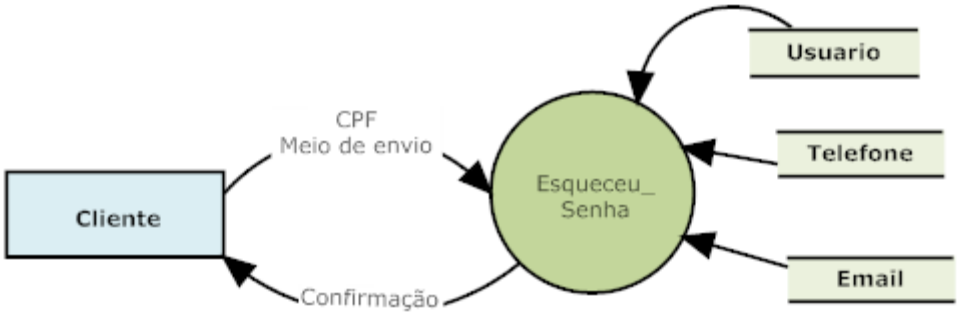


Figura 13 – DFD Esqueceu Senha

Descrição do processo primitivo:

- *Esqueceu_Senha*
    - Objetivos: Enviar uma nova senha ao cliente.

A função recebe o CPF do cliente e o meio de envio desejado. Verifica se o usuário e seus dados, telefone e e-mail, existem e envia uma nova senha ao usuário.

- Procurar produtos

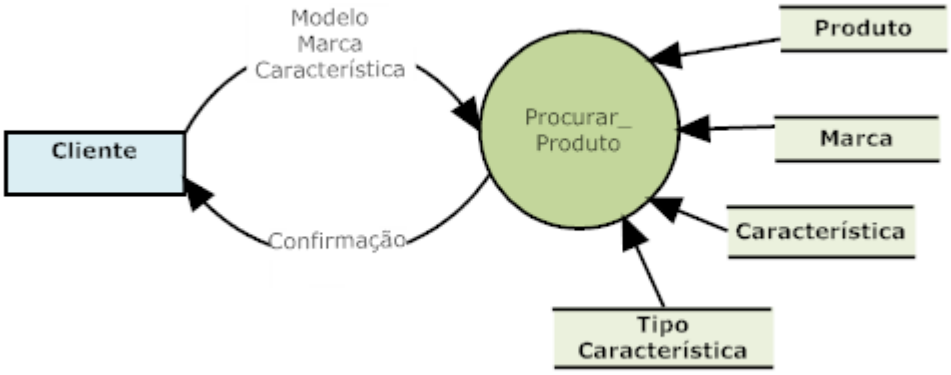


Figura 14 – DFD Procurar produtos

Descrição do processo primitivo:

- *Procurar_Produto*

    - Objetivos: Realizar a busca de um produto.

A função recebe os parâmetros e realiza a consulta. Se esta tiver sucesso, envia os resultados para o usuário. Senão existirem produtos cadastrados, retorna uma mensagem dizendo que o produto não existe.

- Adicionar produto ao carrinho

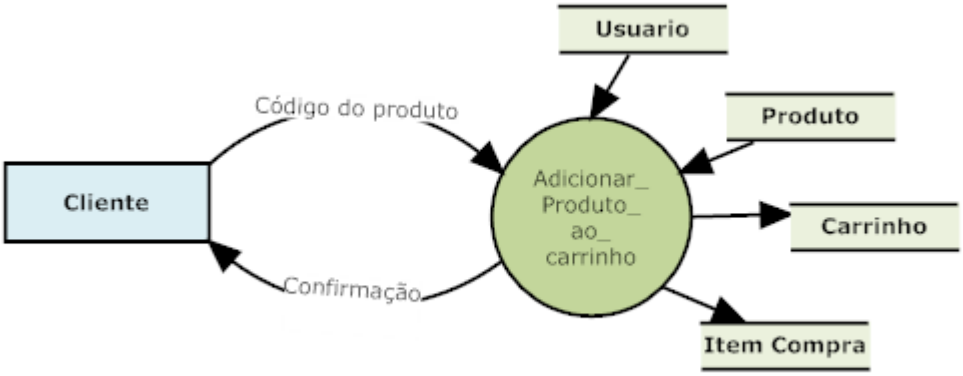


Figura 15 – DFD Adicionar produto ao carrinho

Dada a complexidade da função Adicionar_produto_ao_Carrinho, foi necessário explodi-la.

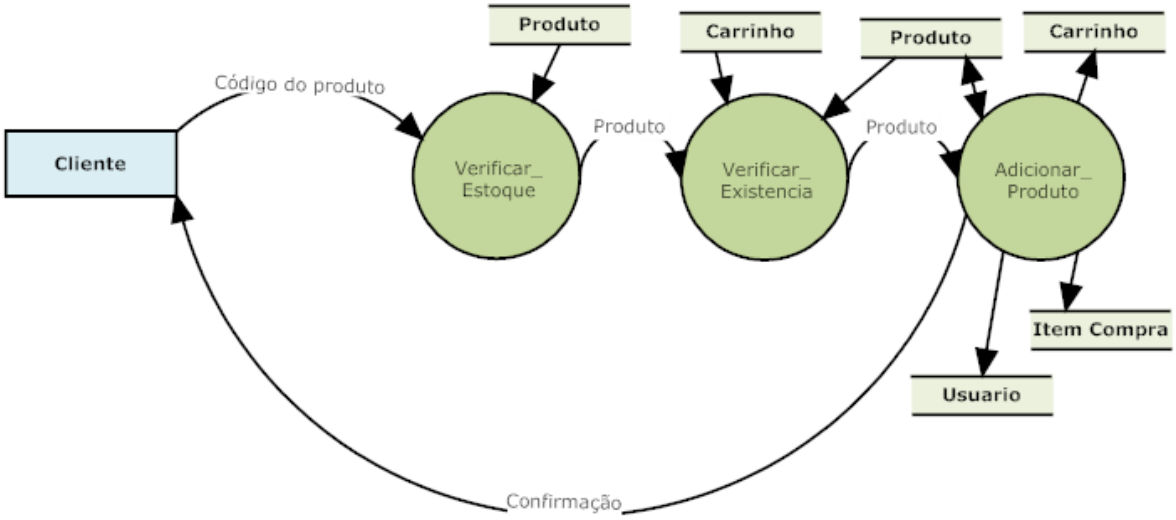


Figura 16 – DFD explodida de Adicionar produto ao carrinho

Especificação dos processos primitivos:

- *Verifica_Estoque*

    - Objetivos: Verifica se a quantidade do produto especificada existe no estoque.

Esta função recebe o parâmetro produto. Com o produto, procura no depósito Produto quantos produtos ainda existem no estoque.
Se o valor do estoque for igual a zero, retorna erro para o cliente.
Senão, repassar o parâmetro recebidos para a próxima função.

- *Verifica_Existencia*

    - Objetivos: Verifica se o produto já está no carrinho ou não.

Recebe o parâmetro produto. Com o parâmetro produto faz uma consulta ao Carrinho:
Se existir no carrinho, colocar Existência com o valor já adicionado ao carrinho.
Senão, colocar com Existência com zero.
Em seguida, envia os dados recebidos e o gerado para a próxima função.

- *Adiciona_Produto*

    - Objetivos: Adicionar o produto ao carrinho.

Recebe o parâmetro produto e existência da função anterior.
Se existência for diferente de zero, edita a quantidade já existente no carrinho, adicionando a ela o valor de existência.
Senão, adiciona o produto ao carrinho.
Em seguida, para ambos os casos, envia uma confirmação ao cliente.

- Ver carrinho

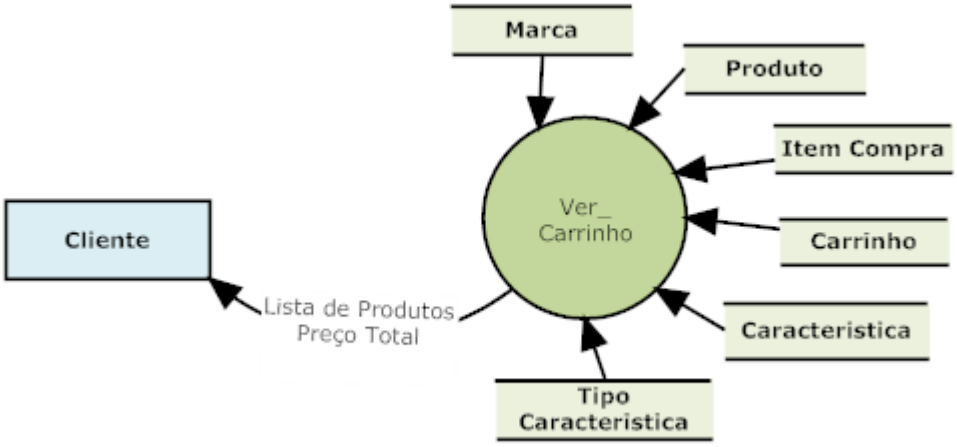


Figura 17 – DFD Ver carrinho

Descrição do processo primitivo:

- *Ver_Carrinho*
  - Objetivos: Consultar os produtos do carrinho do cliente.

A função recebe a identificação do cliente e verifica que produtos estão no carrinho do respectivo usuário, calculando o preço total dos produtos no carrinho.

- Excluir produto do carrinho

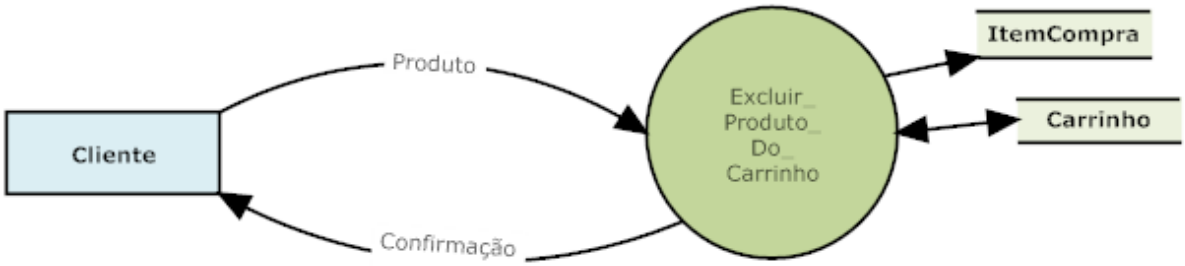


Figura 18 – DFD Excluir produto do carrinho

Descrição do processo primitivo:

- *Excluir_Produto_Carrinho*
    - Objetivos: Excluir produtos do carrinho do cliente.

A função recebe o código do produto, exclui do carrinho e envia uma confirmação para o cliente.

- Alterar quantidade de produtos no carrinho

Figura 19 – DFD Alterar quantidade de produtos no carrinho

Descrição do processo primitivo:

- *Alterar_Quantidade_de_produtos_Carrinho*

- Objetivos: Alterar quantidade de produtos do carrinho do cliente.

A função recebe o produto que se deseja excluir do carrinho, edita o carrinho e envia uma confirmação ao cliente.

- Adicionar produto aos favoritos

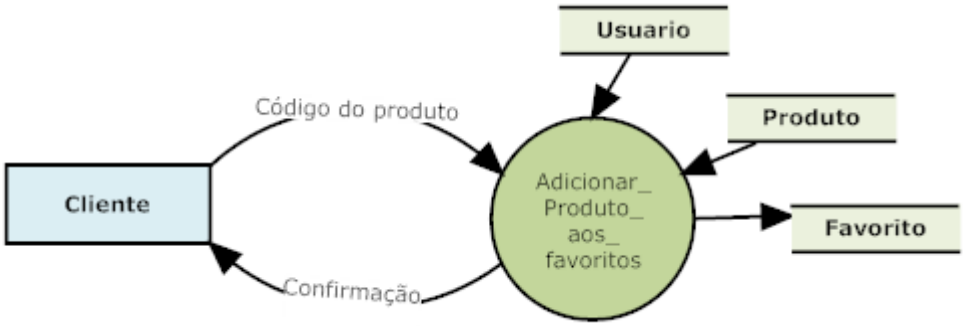


Figura 20 – DFD Adicionar produto ao favorito

- *Adiciona_Produto_aos_Favoritos*

  - Objetivos: Adicionar o produto ao favorito.

  Recebe o parâmetro produto.
  Se existência for diferente de zero, adiciona produto ao favorito.
  Em seguida envia uma confirmação ao cliente.

- Ver favorito

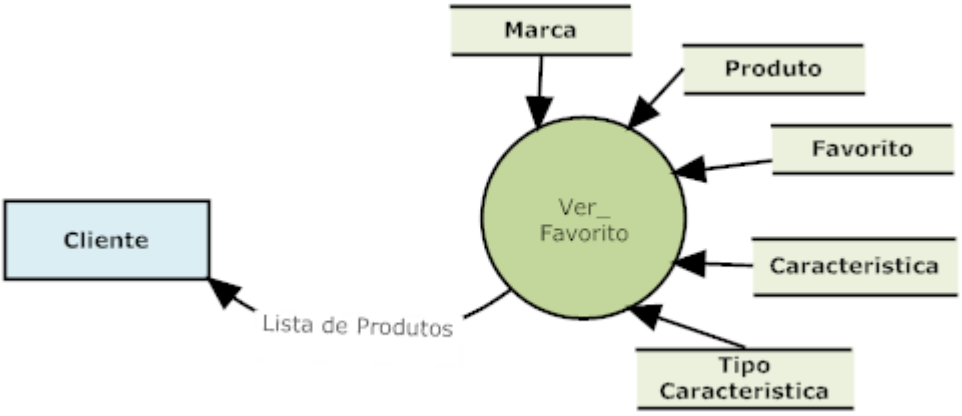


Figura 21 – DFD Ver favorito

Descrição do processo primitivo:

- *Ver_Favorito*
    - Objetivos: Consultar os produtos dos favoritos do cliente.

A função recebe a identificação do cliente e verifica que produtos estão nos favoritos do respectivo usuário.

- Excluir produto do favorito

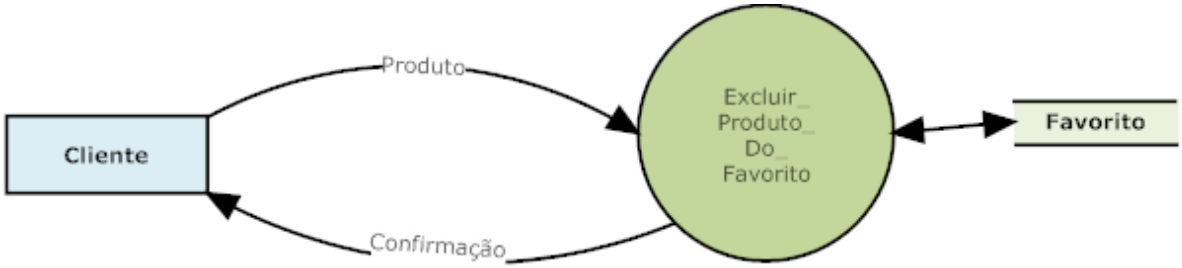


Figura 22 – DFD Excluir produto do favorito

Descrição do processo primitivo:

- *Excluir_Produto_Do_Favorito*
    - Objetivos: Excluir produtos dos favoritos do cliente.

A função recebe o código do produto, exclui dos favoritos e envia uma confirmação para o cliente.

- Alterar quantidade de produtos no favorito

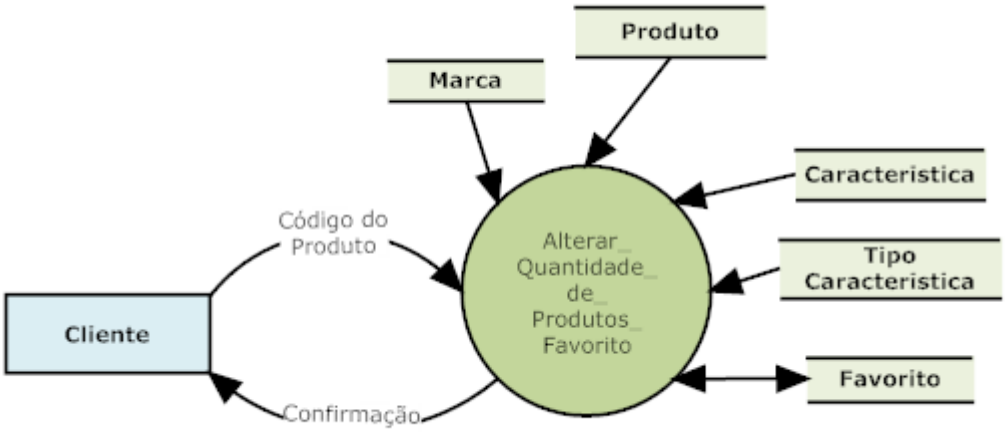


Figura 23 – DFD Alterar quantidade de produtos no favorito

Descrição do processo primitivo:

- *Alterar_Quantidade_de_produtos_Favorito*
    - Objetivos: Alterar quantidade de produtos do favorito do cliente.

A função recebe o produto que se deseja excluir do favorito, edita o favorito e envia uma confirmação ao cliente.

- Efetivar Compra

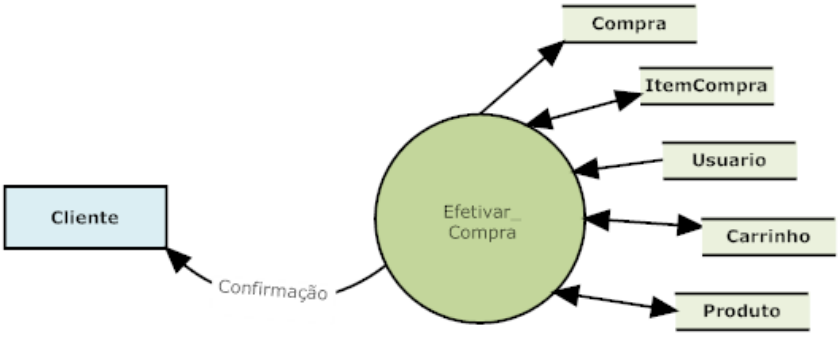


Figura 24 – DFD Efetivar compra

Descrição do processo primitivo:

- *Efetivar_Compra*
    - Objetivos: Efetuar a compra do cliente

A função recebe a identificação do cliente e dos produtos no carrinho do mesmo e efetiva a compra desejada. Uma vez que a compra é efetuada, o carrihno do cliente é esvaziado e a quantidade do produto no estoque é atualizada.

- Alterar Senha

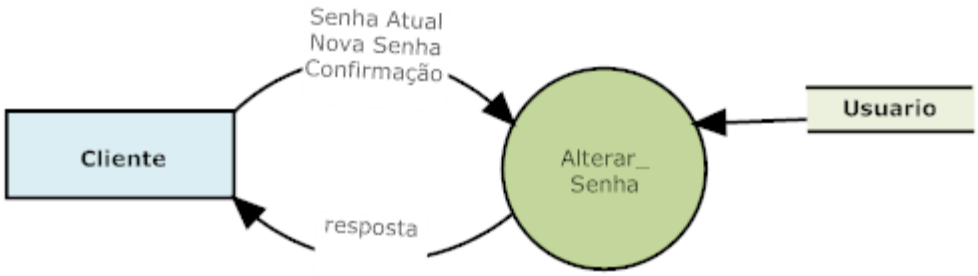


Figura 25 – DFD Alterar senha

Descrição do processo primitivo:

- Alterar_Senha:
  - Objetivos: Modificar a senha do cliente.

A função recebe a senha desejada, verifica se é válida e retorna uma mensagem de sucesso caso a mesma seja válida.

4.1.3.3 DFDs para administrador

- Fazer Login Administrador

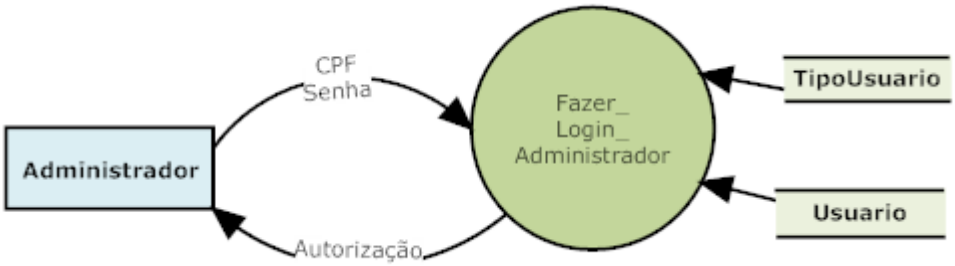


Figura 26 – DFD Login do administrador

Descrição do processo primitivo:

- Fazer_Login_Administrador:
  - Objetivos: Autenticar o administrador, permitindo que ele realize as operações de administração.

A função recebe o CPF e a senha criptografada do administrador e consulta "Usuario" e "TipoUsuario" para saber se elas são compatíveis.

Se o CPF não estiver na base de dados, retorna mensagem de usuário não cadastrado.

Se a senha for correta, é enviada uma autorização para ele.
Senão, é enviada uma mensagem de acesso não autorizado.

- Fazer Logoff Administrador

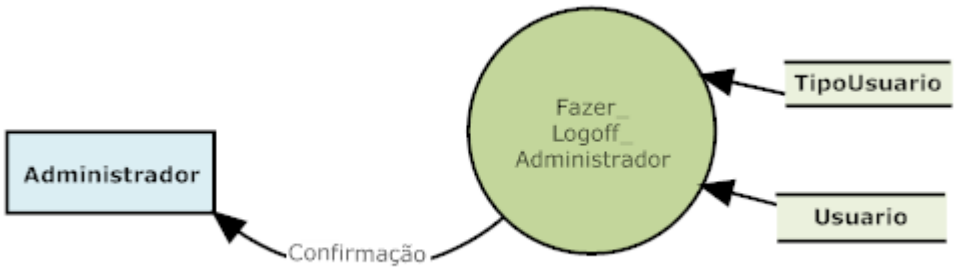


Figura 27 – DFD Logoff do administrador

Descrição do processo primitivo:

- *Fazer_Logoff_Administrador*
  - Objetivos: Desconectar o administrador do site.

Quando for recebido um pedido de logoff, a função irá desconectar o administrador, e enviará uma mensagem de operação realizada com sucesso.

- Cadastrar de marcas

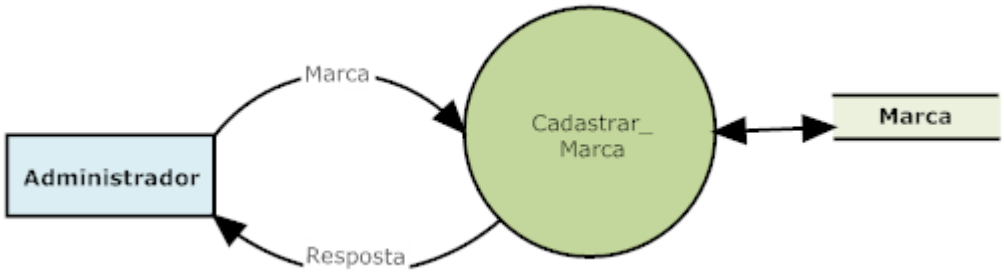


Figura 28 – DFD Cadastrar marca

Descrição do processo primitivo:

- *Cadastrar_Marca*
  - Objetivos: Cadastrar marcas de produtos no sistema.

Recebe os parâmetros, inclui uma nova marca nos depósitos de dados, e envia ao terminador uma mensagem de sucesso.

- Editar marca

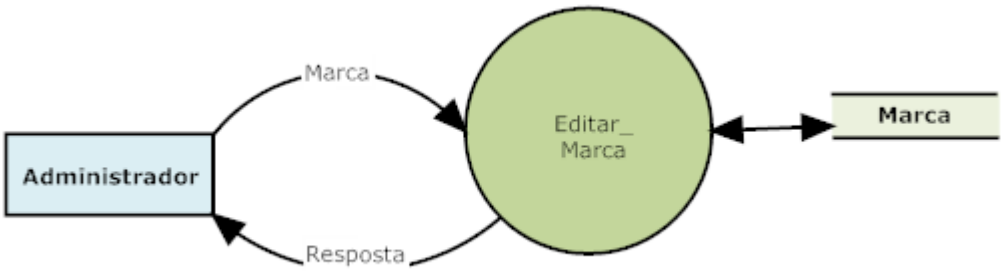


Figura 29 – DFD Edita marca

Descrição do processo primitivo:

- *Edita_Marca*
  - Objetivos: Editar marcas de produtos no sistema.

Altera na base de dados os dados editados pelo administrador e depois envia uma confirmação para ele.

- Excluir marca

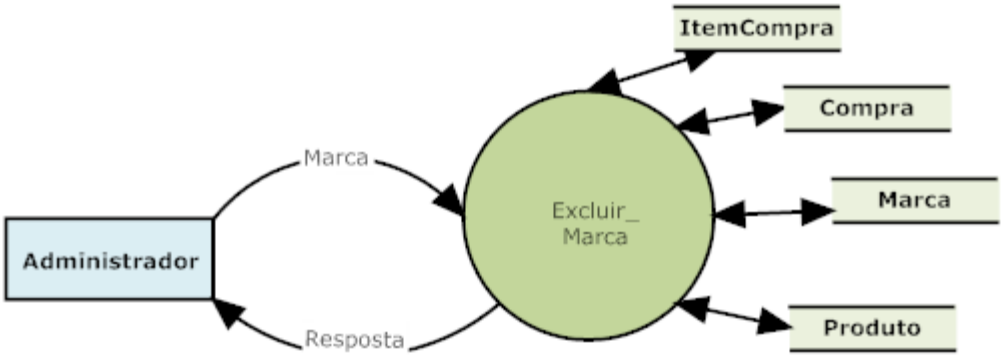


Figura 30 – DFD Excluir marca

Descrição do processo primitivo:

- *Excluir_Marca*
    - Objetivos: Excluir marcas e todos os produtos referenciados a respectiva marca no sistema.

Exclui a marca e os produtos referentes a marca. Retorna uma mensagem de sucesso caso a marca tenha sido excluída com sucesso.

- Cadastrar produto

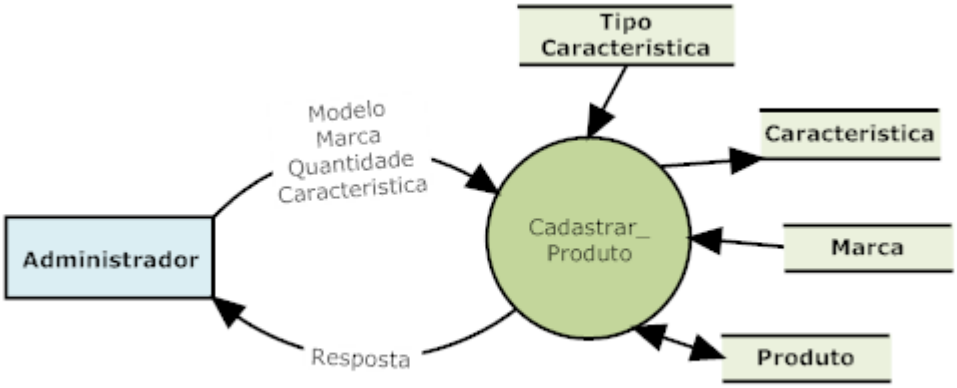


Figura 31 – DFD Cadastrar Produto

Descrição do processo primitivo:

- *Cadastrar_Produto*
    - Objetivos: Cadastrar produtos no sistema.

Recebe os parâmetros, inclui o novo produto no depósito de dados, e envia ao terminador uma mensagem de sucesso.

- Editar produto

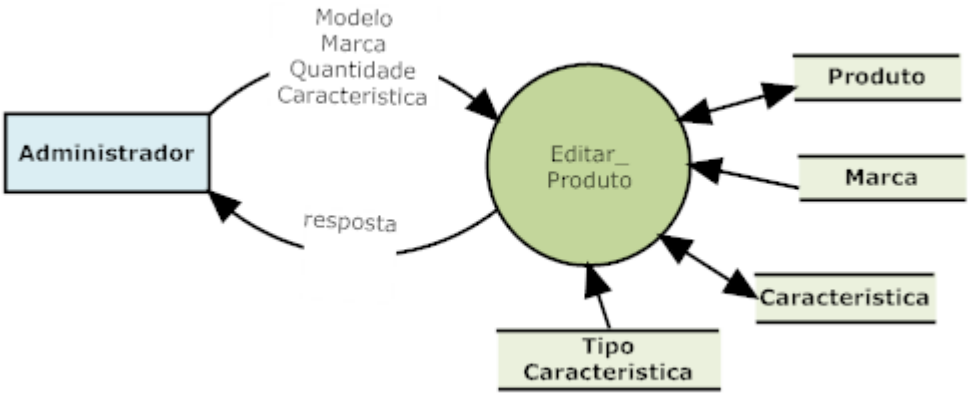


Figura 32 – DFD Editar Produto

Descrição do processo primitivo:

- *Editar_Produto*
  - Objetivos: Editar produtos no sistema.

Altera na base de dados os dados editados pelo administrador e depois envia uma confirmação para ele.

- Excluir produto

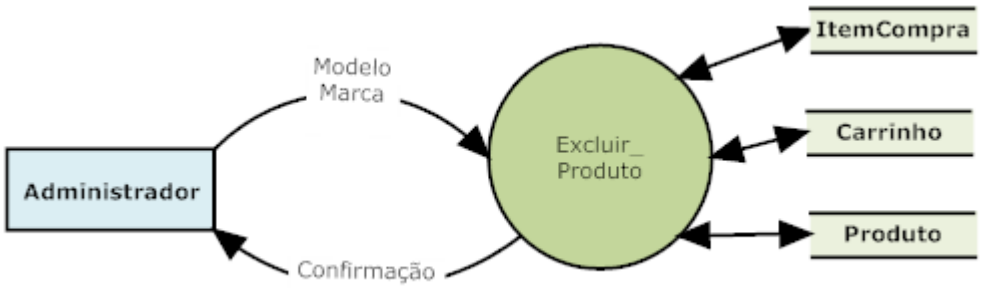


Figura 33 – DFD Excluir Produto

Descrição do processo primitivo:

- *Excluir_Produto*
    - Objetivos: Excluir produtos no sistema.

Exclui os produtos selecionados pelo administrador do sistema. Caso o produto esteja no carrinho de algum cliente, o produto será removido de "Carrinho" e "ItemCompra" . Após a confirmação de exclusão do produto, o sistema retorna uma mensagem de sucesso caso o produto tenha sido excluído com sucesso.

No caso de o produto pertencer a alguma compra, ele será excluído da mesma maneira, logo ele não fará mais parte da respectiva compra.

- Alterar Senha Administrador

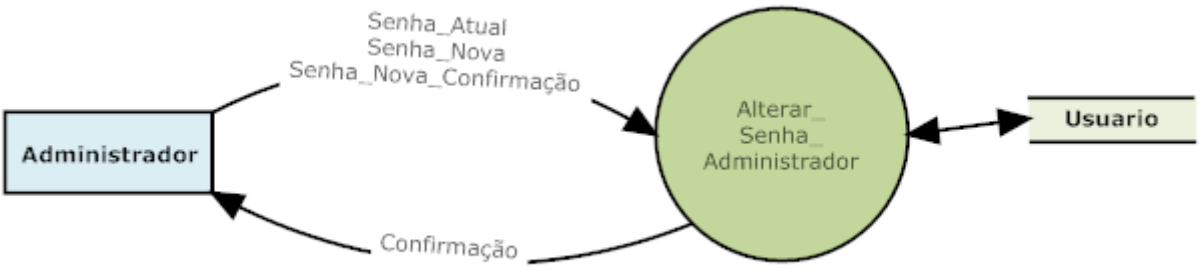


Figura 34 – DFD Alterar Senha do administrador

Descrição do processo primitivo:

- *Alterar_Senha_Administrador*
    - Objetivos: Altera a senha do administrador.

Recebe a senha atual e verifica se a mesma está correta. Caso positivo, a senha é alterada se a confirmação da senha for igual a nova senha.

- Esqueceu Senha

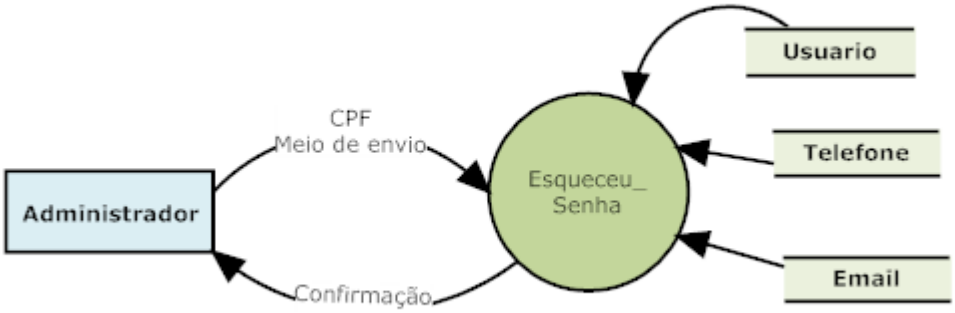


Figura 35 – DFD Esqueceu senha

Descrição do processo primitivo:

- *Esqueceu_Senha*
    - Objetivos: Enviar uma nova senha ao administrador.

A função recebe o CPF do cliente e o meio de envio desejado. Verifica se o usuário e seus dados, telefone e e-mail, existem e envia uma nova senha ao usuário.

- Cadastrar Alerta

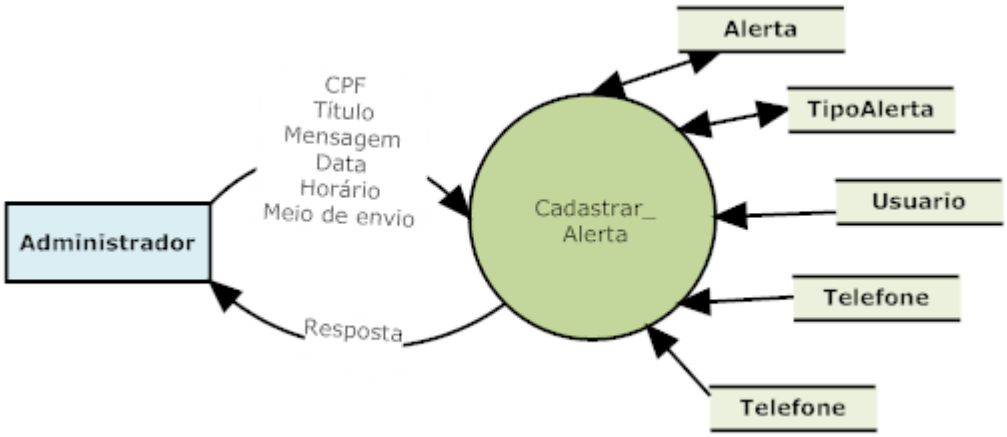


Figura 36 – DFD Alterar Senha do administrador

Descrição do processo primitivo:

- *Cadastrar_Alerta*

- Objetivos: Cadastrar algum alerta para o cliente como por exemplo a chegada de novos produtos.

A função recebe CPF, título da mensagem, data da mensagem, horário da mensagem e o meio de envio (e-mail ou *sms*). Verifica se os parâmetros são válidos e agenda um alerta para ser enviado ao cliente.

- Cadastrar Característica

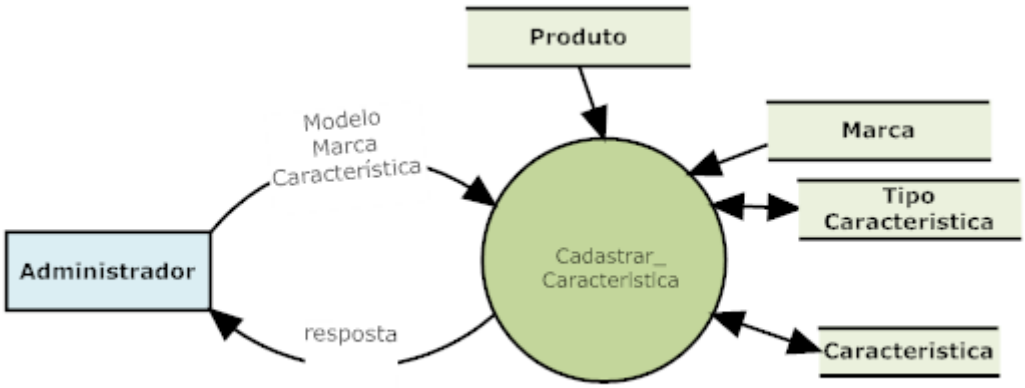


Figura 37 – DFD Cadastrar Característica

Descrição do processo primitivo:

- *Cadastrar_Caracteristica*
  - Objetivos: Cadastrar características de produtos.

A função recebe o código do produto que se deseja editar as características. Verifica a existência do produto. Caso positivo, cadastra as características do mesmo.

- Editar Característica

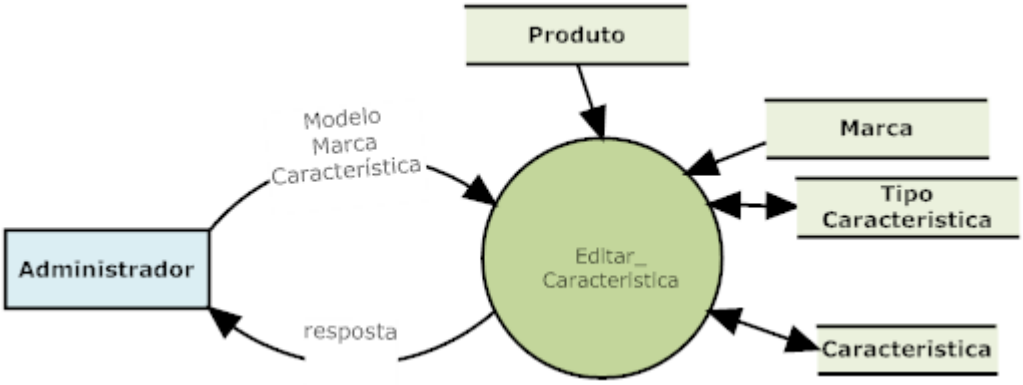


Figura 38 – DFD Editar Característica

Descrição do processo primitivo:

- *Cadastrar_Caracteristica*
    - Objetivos: Cadastrar características de produtos.

A função recebe o código do produto que se deseja editar as características. Verifica a existência do produto. Caso positivo, edita as características do mesmo.

- Procurar Usuário

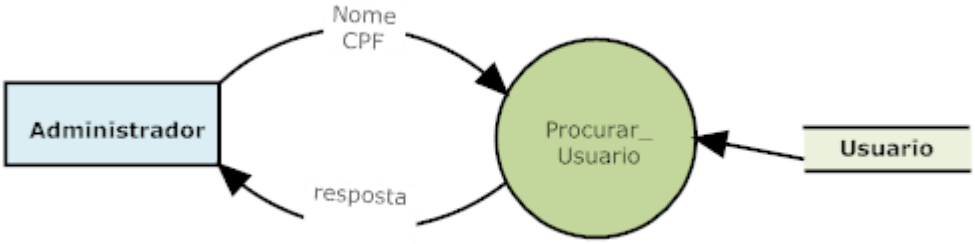


Figura 39 – DFD Alterar Senha do administrador

Descrição do processo primitivo:

- *Procurar_Usuario*
    - Objetivos: Procurar um usuário que esteja cadastrado no sistema.

A função recebe o nome ou o CPF do usuário, verifica a sua existência no banco de dados e retorna o usuário com o respectivo CPF ou os usuários com o respectivo nome.

4.1.4 Diagrama de Transição de Estado

O Diagrama de Transição de Estados (DTE) é necessário para expressar uma visão do comportamento do sistema em relação aos seus diferentes estados válidos.

O DTE do sistema foi separado em três DTE's distintos, um para o administrador, outro para o cliente e outro para o visitante. Esses DTE's mostram a dependência de cada processo em relação ao outro, mostrando em que ordem precisam ser executados. Para qualquer tipo de usuário, o estado para sair do sistema, é possível de ser solicitado em qualquer momento.

- **DTE para o Cliente**

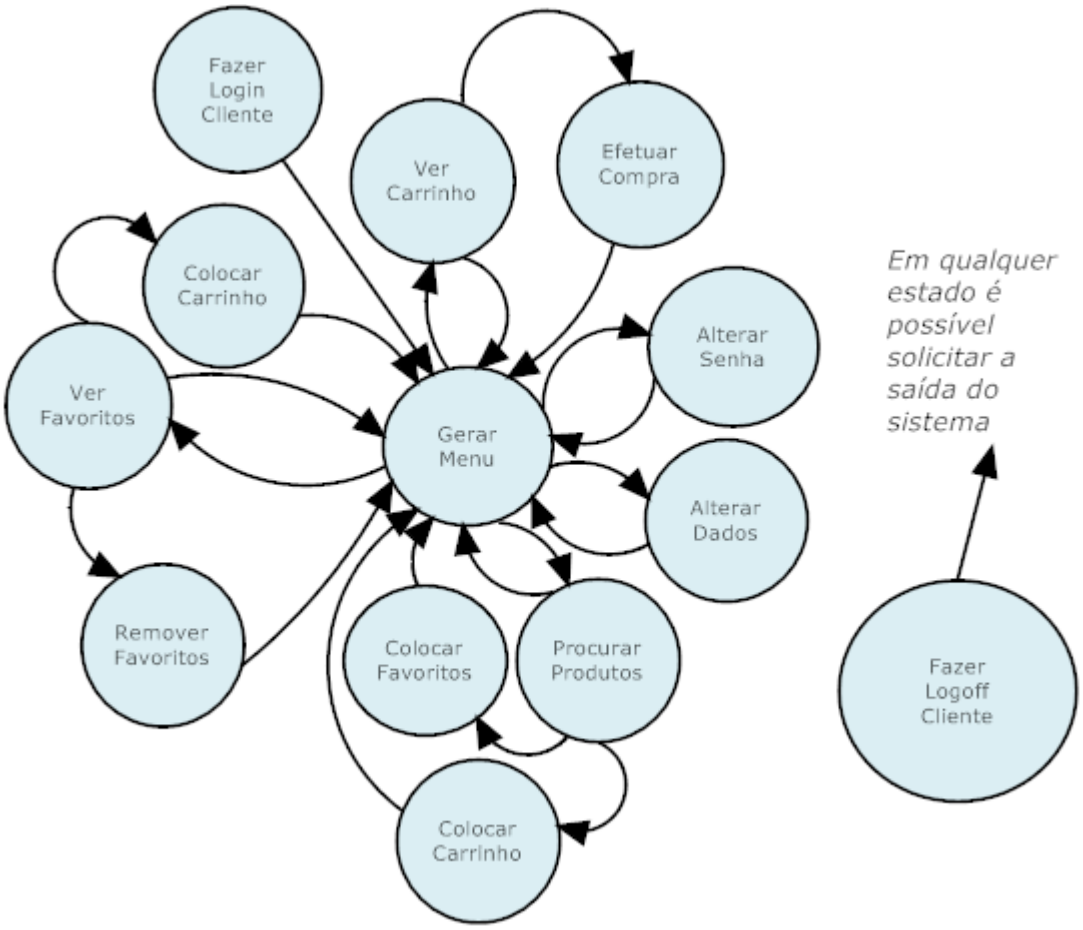


Figura 40 – Diagrama de Transição de Estado para o cliente

Uma vez autenticado no sistema, o cliente poderá realizar todas as ações acima. Em qualquer estado ele poderá pedir o logoff do sistema, sendo este um estado terminal que o levará para a página principal da loja.

- **DTE para o Administrador**

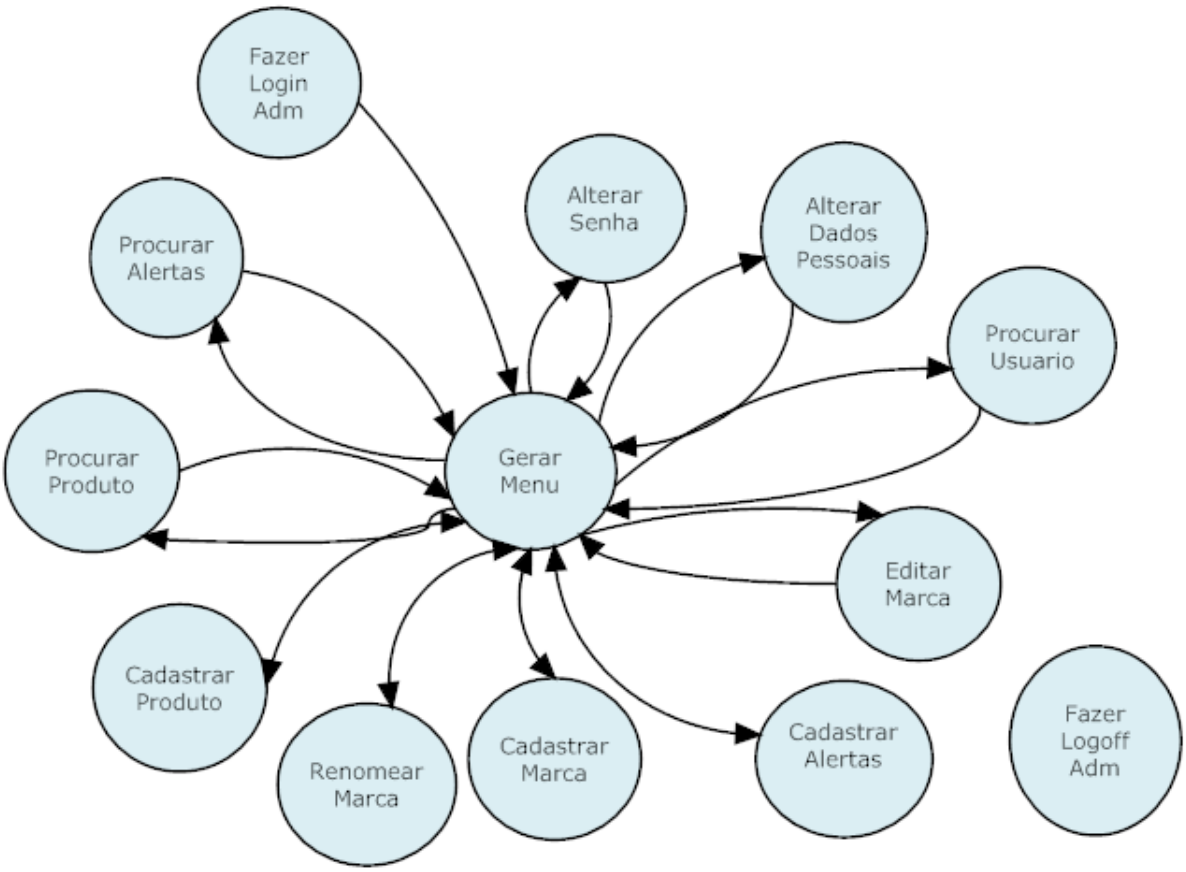


Figura 41 – Diagrama de Transição de Estado para o administrador

O DTE acima mostra as tarefas que o administrador pode realizar no sistema. No momento em que é feita a autenticação, o admnistrador pode realizar qualquer uma das opções mostradas acima. É importante ressaltar que em qualquer momento, o administrador poderá fazer o logoff do sistema. Este estado é mostrado acima isoladamente, porém ele é um estado terminal que pode ser acessado por qualquer outro estado.

- **DTE para o Visitante**

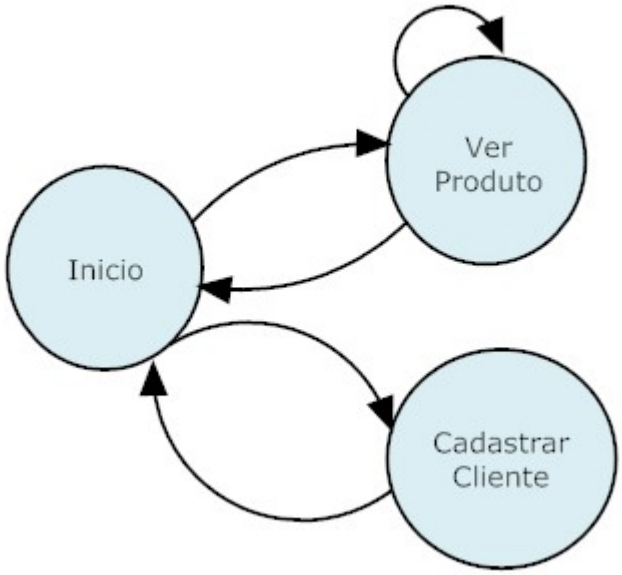


Figura 42 – Diagrama de Transição de Estado para o administrador

No DTE do visitante, o mesmo poderá ficar consultando os produtos exixtentes na loja mas não poderá comprá-los. A partir do momento que ele se cadastra e faz o login, ele se torna um cliente e o DTE se torna análogo ao DTE de cliente.

4.1.5 Módulo *GSM*

Além da interface *web*, o sistema se comunicará com o cliente através de mensagens de textos via celular através de uma rede de telefonia celular. Para agendar promoções, novos produtos do sistema ou resolver problemas do usuário como a perda de uma senha, foram utilizados *sockets* para a a comunicação com o servidor.

Um *socket*, por definição, é um canal de comunicação entre computadores em uma rede e identifica uma conexão entre eles, normalmente entre um cliente e um servidor. Através dos *sockets* os computadores podem trocar informações através de uma rede. Para identificar uma conexão ente dois computadores, um *socket* deve ser definido, por meio das seguintes informações:

- Endereço IP do servidor;
- Porta onde se encontra o serviço solicitado;
- Endereço IP do cliente;
- Porta através da qual o cliente solicita o serviço.

Um servidor *web* tem a porta 80 como porta padrão de comunicação entre os clientes. Quando digitamos um endereço de um *site* no Internet Explorer do Windows, automaticamente esse endereço é convertido em seu respectivo endereço IP. Se estamos numa rede, nosso micro tem um único endereço IP. E finalmente, junto deste processo, uma porta em seu computador é disponibilizada dinamicamente, tendo um número maior que 1024, para esta conexão.

Então, temos todas as informações necessárias para estabelecer a conexão, usando assim um *socke*t. O cliente, no caso de uma conexão a uma página da Internet, é quem a solicita através de um *browser* (Internet Explorer, por exemplo), e o servidor é quem disponibiliza a página para ser acessada. No projeto o uso de *sockets* foi feito para envio de e-mail e *sms* (mensagens de texto para celular). Pode-se ver na figura a seguir um esquema de como funciona o envio de *sms*.

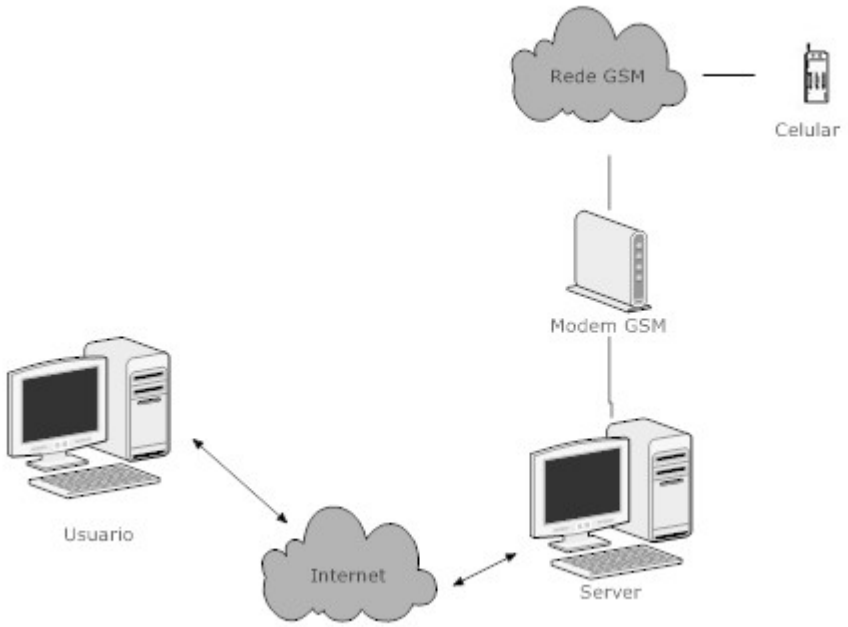


Figura 43 – Módulo *GSM*

Para o envio de torpedos, foi utilizado a biblioteca "gsmlib" que era responsável pela comunicação com o modem *GSM*. Com esta biblioteca instalada em uma máquina com sistema operacional Linux (pode ser Windows também), o usuário pode enviar torpedos pela linha de comando, bastando apenas digitar o celular e a mensagem desejada. A biblioteca é responsável pela comunicação com o modem *GSM* e este por sua vez envia a mensagem.

Para o envio de torpedos foi feito um pequeno programa em C que verifica em uma determinada pasta se há algum *sms* agendado para ser enviado. Dentro da respectiva pasta, existe um arquivo texto que contém as seguintes informações para o envio do torpedo: dia, mês, ano, horário, o número do celular que se deseja enviar a mensagem e a mensagem. Com a existência deste arquivo, o programa pega o número do celular, a mensagem, verifica a hora e dia e envia a mensagem para o respectivo usuário.

Para a comunicação com a operadora de telefonia móvel será utilizado um modem *GSM*. Para o funcionamento do mesmo precisamos ter um chip com créditos e posicioná-lo em uma área com um bom nível de sinal. A comunicação entre o sistema e o modem se dá através de alguns comandos AT. No entanto, como existem bibliotecas que já integram esses comandos, em nosso trabalho nos concentraremos apenas na utilização das mesmas.

4.2 Projeto

4.3 Decomposição em módulos

Nesta parte, o sistema é decomposto em módulos, que apresentam o menor acoplamento possível. O número de módulos é mantido pequeno, para que eles possam apresentar todas as funcionalidades necessárias, sem necessitar de trocas de mensagens constantes com outros módulos.

Abaixo é possível ver o processo de comunicação dos módulos. Através do browser, o usuário poderá fazer um pedido de cadastro, uma consulta ou efetuar compra, este pedido será enviado ao servidor, onde estarão rodando as funções *php*, que realizarão a consulta ou cadastro no banco de dados.

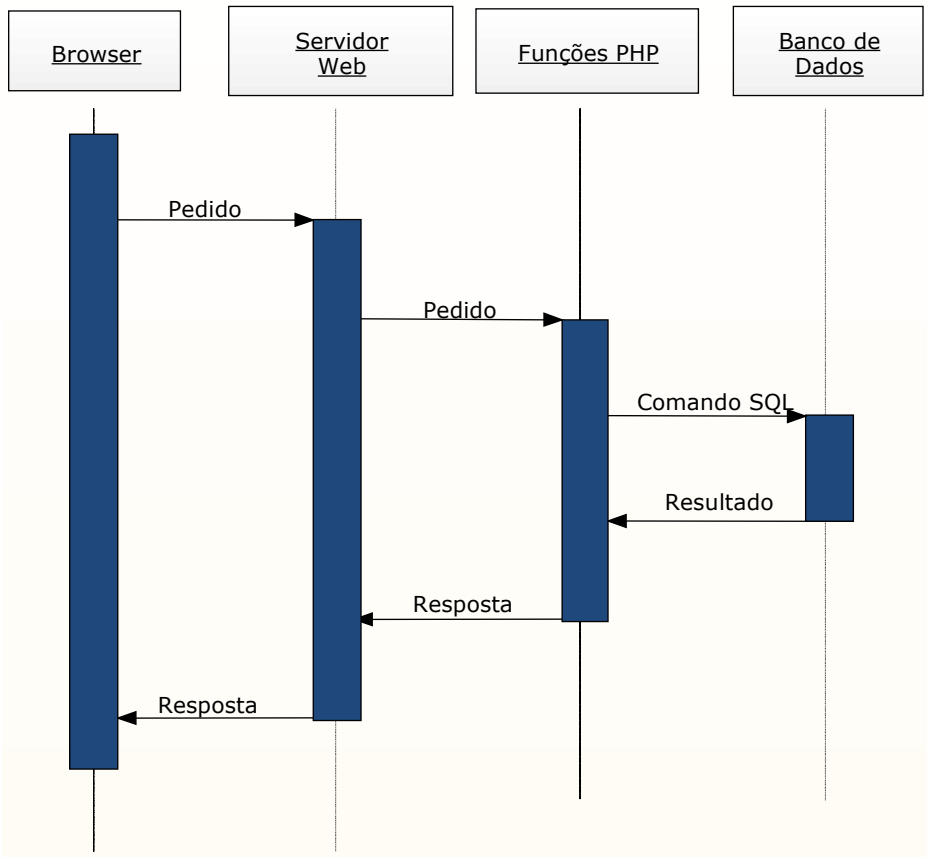


Figura 44 – Diagrama de comunicação entre o browser e o bando de dados

Com a figura abaixo fica mais fácil entender a arquitetura do site e posteriormente os seus módulos.

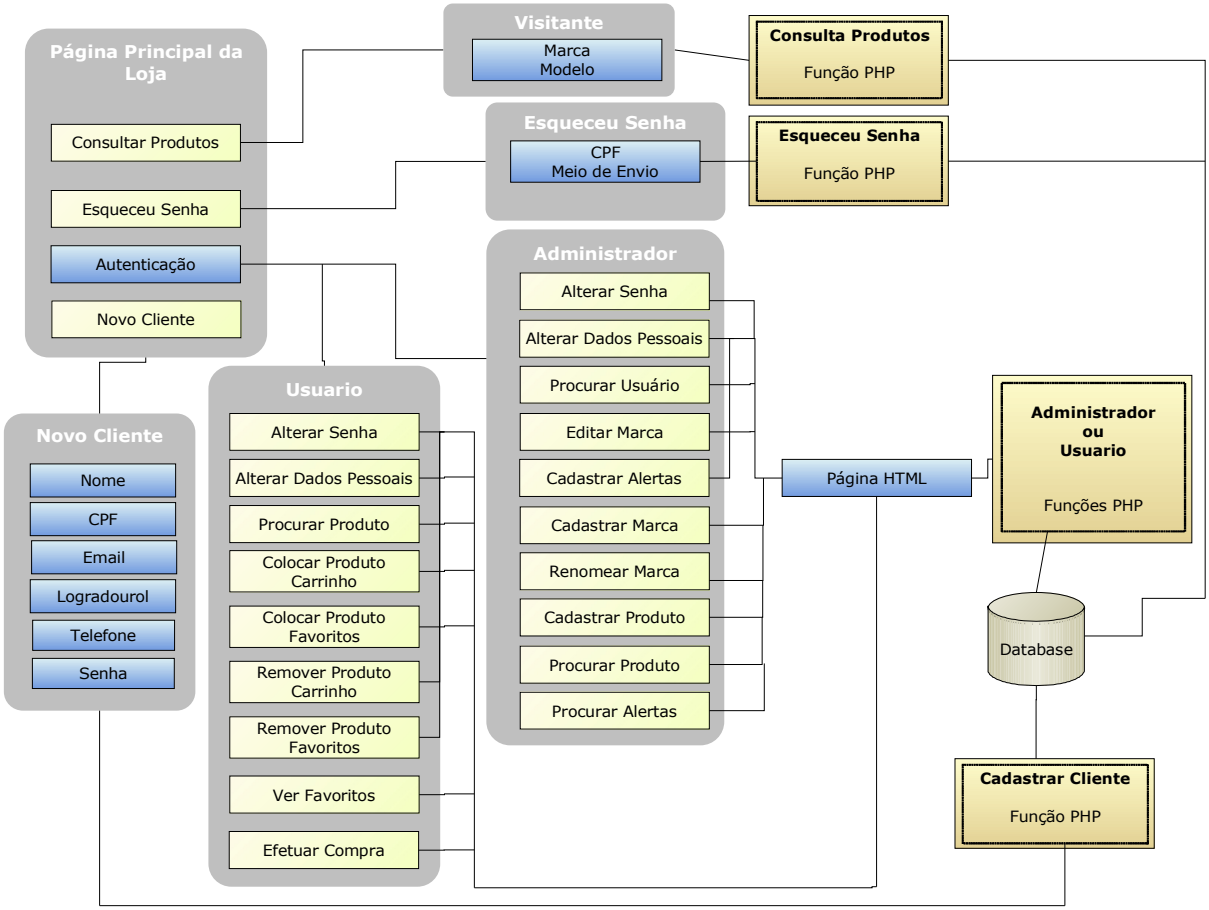


Figura 45 – Arquitetura do site

Na página principal da loja, o usuário poderá autenticar-se no sistema preenchendo os campos CPF e senha. Se não for cliente haverá um link onde ele poderá consultar os produtos da loja. Através do link consultar produtos, ele será levado a uma outra página onde poderá efetuar uma busca preenchendo os campos de marca ou modelo. A respectiva função php fará a busca no banco de dados e retornará o(s) produto(s). Além disto, na página principal, haverá um outro link onde o usuário poderá recuperar a sua senha. Ao clicar no link esqueceu senha, o usuário é levado a uma página *html* em que é necessário preencher os campos CPF e meio de envio. A função *php* correspondente a esta tarefa realizará uma busca no banco de dados para verificar se o respectivo usuário existe e coletar seus dados para enviar uma nova senha.

Uma vez autenticado no sistema, será gerado o respectivo menu, cliente ou administrador, onde o usuário poderá realizar as operações oferecidas pelo sistema. Ao escolher uma

operação, é gerado uma página *html* onde o usuário entre com os dados e a respectiva função *php* realiza a comunicação com o banco de dados.

4.3.1 Relações Cadastrais

Esse módulo é responsável pelas operações de cadastro de produtos, cadastro de alertas, cadstro de clientes e cadastro de marcas.

A interface com o usuário é feita através de uma página em *html*. Ao fazer o login o código *php* será responsável por tratar os dados digitados pelo usuário enviando os mesmos ao banco de dados para verificar a sua existência. Todos os outros módulos funcionam de maneira análoga a este.

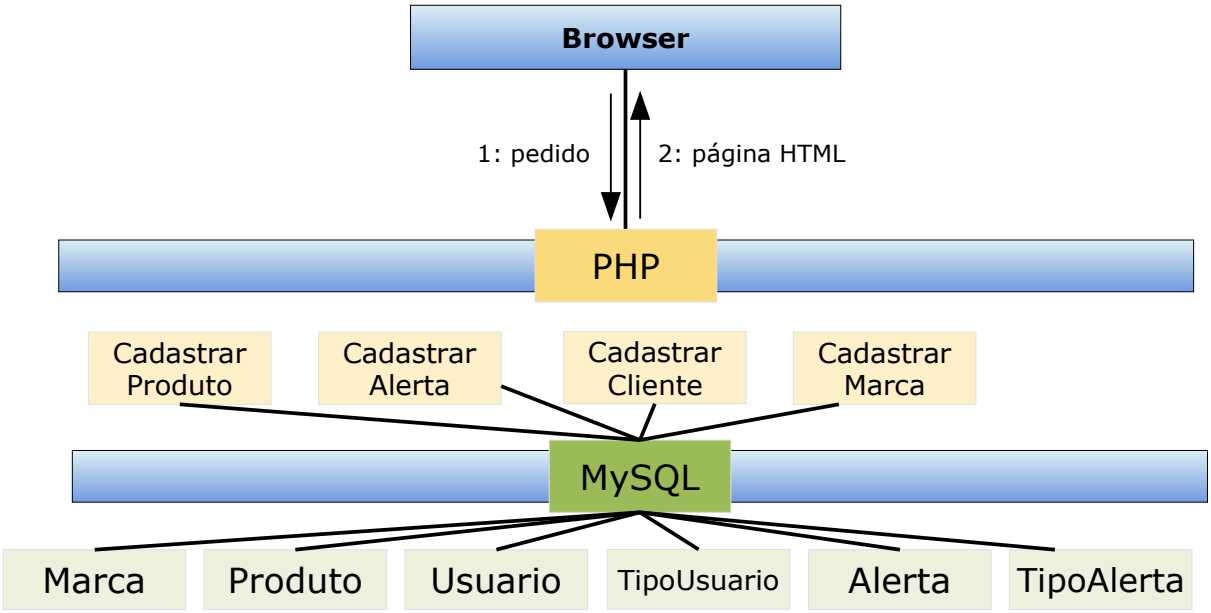


Figura 46 – Relação cadastral

4.3.2 Relações de edição

Responsável pelas operações de editar dados dos usuários, produtos e marcas. De maneira análoga às relações cadastrais a interface é feita através de um arquivo *html*, que chama as funções em *php* para fazer o tratamento dos dados e verificar a validade dos mesmos para poder realizar as ações no banco de dados.

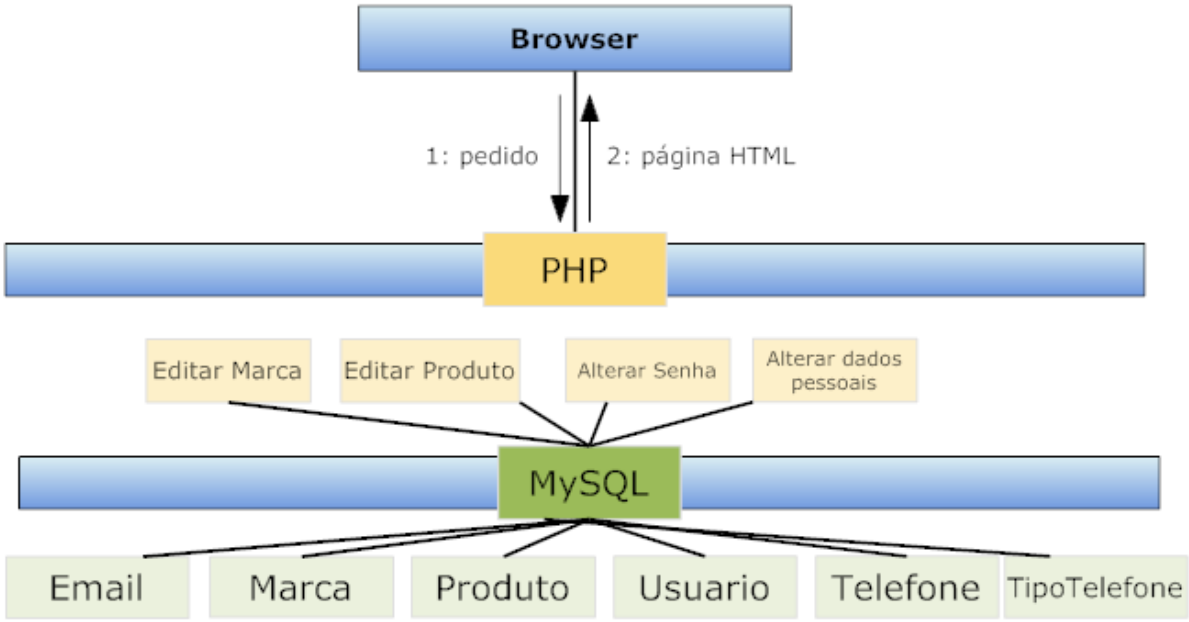


Figura 47 – Relações de edição

4.3.3 Relações Consulta

Realiza a busca de produtos, consulta ao carrinho e a compra do cliente. Funciona de maneira análoga aos módulos descritos acima. A única diferença é que os códigos em *php* realizam apenas consultas no banco de dados.

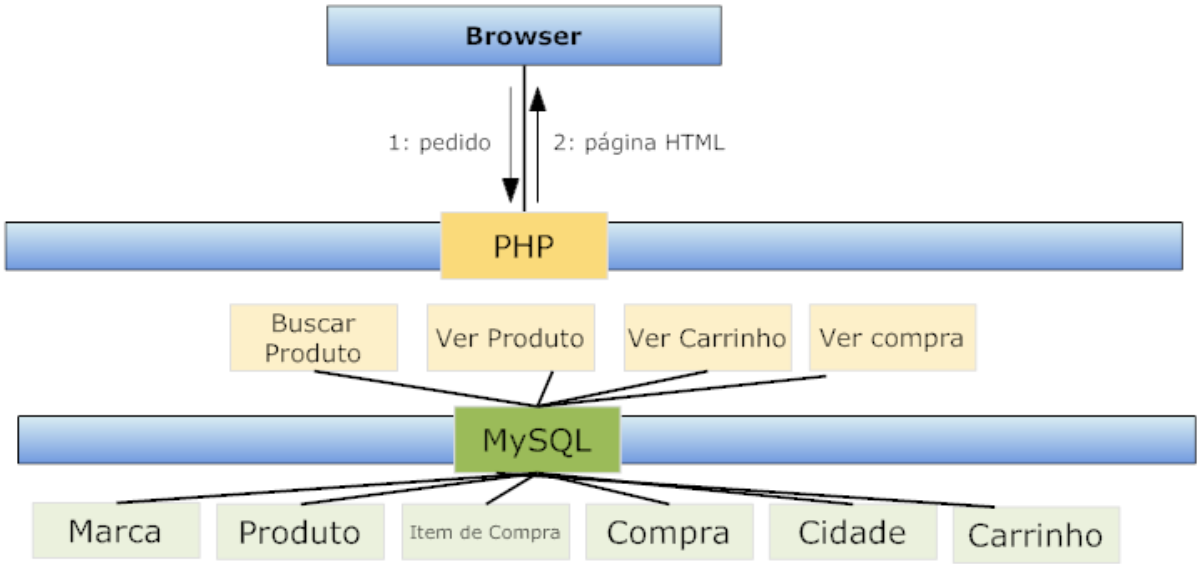


Figura 48 – Relações de consulta

4.3.4 Relações de Produto

Módulo responsável pelas ações que podem ser feitas sobre os produtos do sistema. Ao efetuar a compra a função respectiva em *php* faz a atualização do estoque.

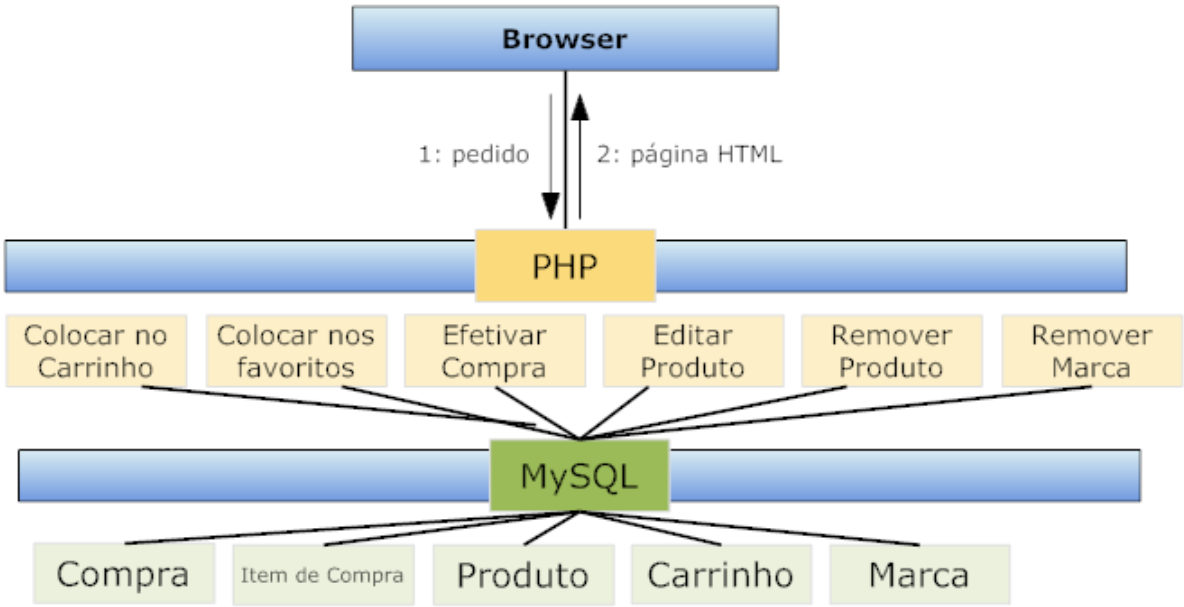


Figura 49 – Relações de produto

4.3.5 Gerar_Menu

O sistema identifica o usuário através de um código que indica se ele é administrador ou cliente, e gera o menu de acordo com este código.

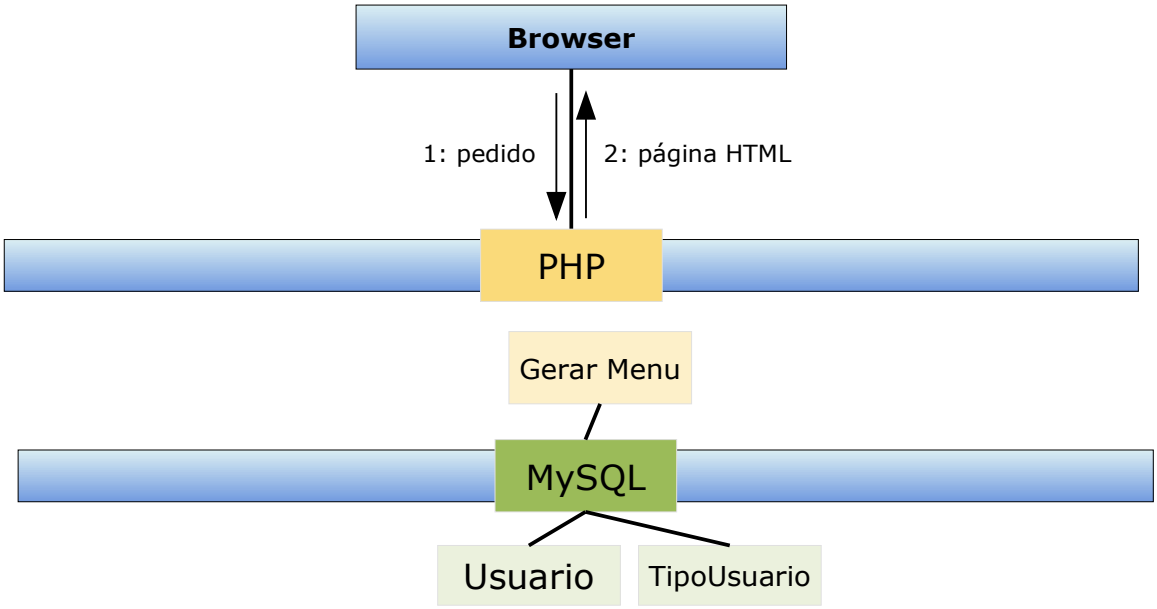


Figura 50 – Relações de geração de menu

4.4 Decomposição em processos concorrentes

Os únicos processos concorrentes que podem ocorrer são no acesso ao banco de dados; um exemplo é o momento em que dois clientes realizam uma compra simultaneamente. A ferramenta de banco de dados escolhida (MySQL) se encarrega de tratar estes casos. O mais comum é que o gerenciador deixe um dos processos em espera, até que a operação seja completada.

Assim, não é necessário se preocupar com processos concorrentes, pois estes serão tratados pelo gerenciador de banco de dados.

4.5 Decomposição de dados

A seguir segue o modelo físico das entidades de dados, que correspondem às tabelas do banco de dados. Pode-se ver com clareza as dependências entre os dados das entidades.

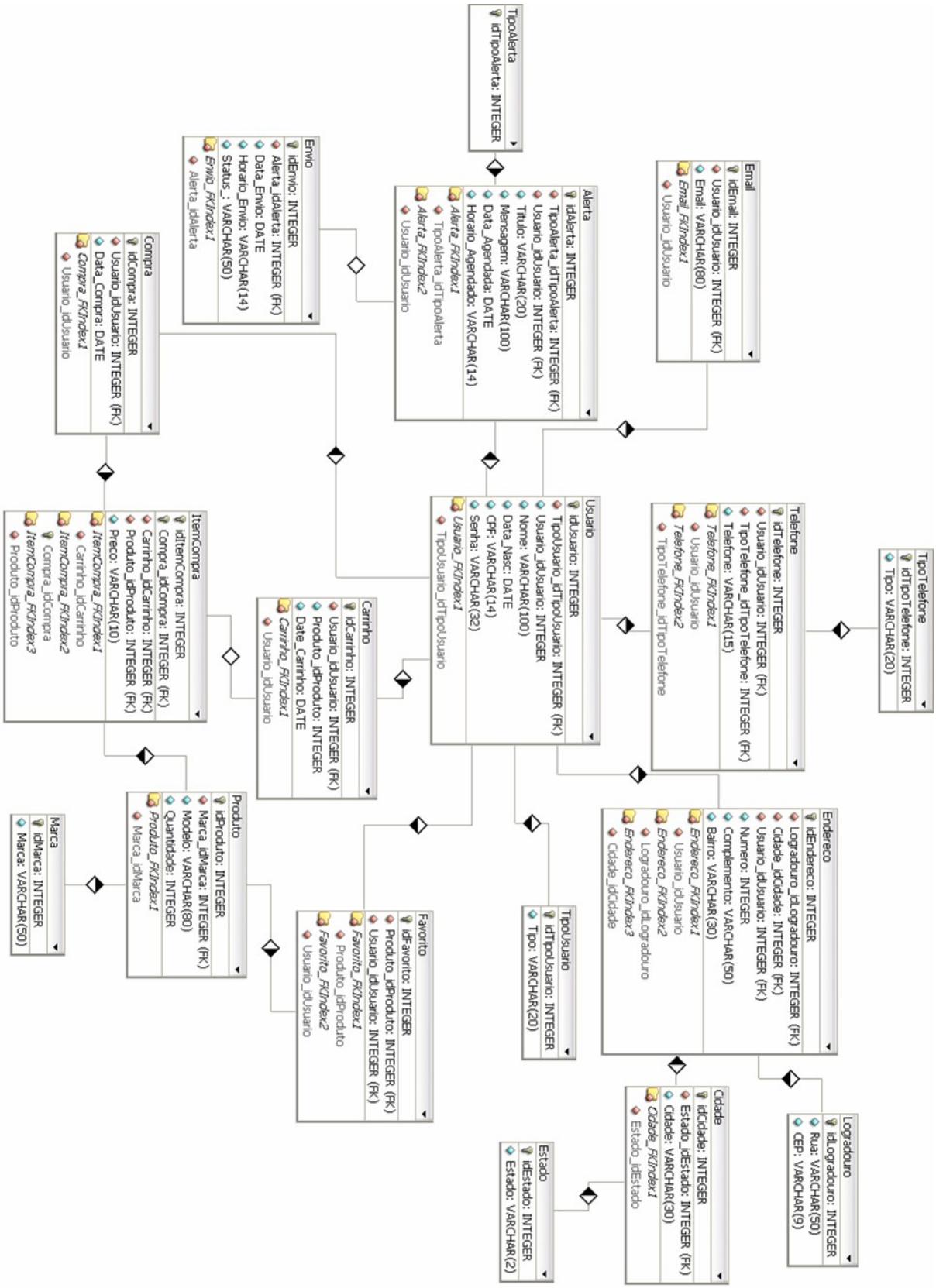


Figura 51 – Diagrama de entidade e relacionamentos do modelo físico

Para o DER temos o seguinte dicionário de dados:

- **Produto –** Entidade que define dados relativos ao produto.
    - **idProduto–** Código que identifica o produto.
    - **Marca_idMarca –** Código da marca do produto.
    - **Modelo –** Modelo do produto.
    - **Quantidade –** Quantidade do produto no estoque.
    - **Marca_idMarca –** Código que identifica a marca do produto.

- **Marca -** Define os vários tipos de marcas do produto.
    - **idMarca –** Código que identifica a marca do produto.
    - **Marca –** Nome da marca do produto.

- **Favorito –** Lista de todos os produtos favoritos do usuário.
    - **idFavorito -** Código que identifica o favorito.
    - **Produto_idProduto -** Código que identifica o produto favorito.
    - **Usuário_idUsuario -** Código que identifica o usuário.
    - **Produto_idProduto –** Código que identifica o produto
    - **Usuário_idUsuario –** Código que identifica o usuário

- **Carrinho –** Entidade que define as compras do usuário.
    - **idCarrinho -** Código que identifica o carrinho.
    - **Data_Carrinho –** Data de compra dos produtos.
    - **Usuário_idUsuario –** Código que identifica o usuário.

- **ItemCompra –** Contém a informação de todos as compras realizadas..
    - **idItemCompra -** Código que identifica o item de compra.
    - **Preço –** Preço da compra.
    - **Carrinho_idCarrinho –** Código que identifica o carrinho.
    - **Compra_idCompra –** Código que identifica a compra.
    - **Produto_idProduto –** Código que identifica o produto.

- **Compra –** Guarda informações relativas a compra**.**
    - **idCompra -** Código que identifica uma compra do usuário.
    - **Data_Compra –** Data em que a compra foi realizada**.**
    - **Usuário_idUsuario –** Código que identifica o usuário.

- **Usuario –** Guarda informações relativas ao usuário.
    - **Id_Usuario -** Código que identifica o usuário.
    - **Nome -** Nome do usuário.
    - **Data_Nasc –** Data de nascimento do usuário**.**
    - **CPF –** CPF do usuário.
    - **Senha** – Senha do usuário**.**
    - **TipoUsuario_idTipoUsuario –** Código que identifica o tipo de usuário.

- **TipoUsuario –** Guarda informação do tipo de usuário (Cliente ou administrador).
    - **idTipoUsuario -** Código que identifica o tipo de usuário.
    - **Tipo –** Guarda a informação sobre o tipo de usuário (Cliente ou administrador).

- **Telefone –** Guarda informações relativas ao telefone do usuário.
    - **idTelefone -** Código que identifica o telefone do usuário.
    - **Telefone –** Guarda o telefone do usuário.
    - **Usuário_idUsuario –** Código que identifica o usuário.
    - **TipoTelefone_idTipoTelefone –** Código que identifica o telefone.

- **TipoTelefone** - Guarda informações relativas ao tipo de telefone do usuário, como por exemplo: telefone comercial, telefone residencial e celular.
    - **idTipoTelefone -** Código que identifica o tipo de telefone.
    - **Tipo** – Nome do tipo de telefone. Pode ser, por exemplo, residencial, comercial, celular, etc.

- **Alerta -** Lista as formas possíveis de alerta (e-mail ou *sms*).
    - **idAlerta -** Código que identifica o alerta.
    - **Título –** Título do alerta que será encaminhado.
    - **Mensagem –** Mensagem que será enviada ao usuário**.**
    - **Data_Agendada –** Data em que o alerta será enviado**.**
    - **Horário_Agendado –** Horário em que o alerta será enviado.
    - **TipoAlerta_idTipoAlerta –** Código que identifica o tipo de alerta.
    - **Usuário_idUsuario –** Código que identifica o usuário.

- **TipoAlerta -** Apresenta o tipo de alerta.
    - **idTipoAlerta -** Código que identifica o tipo de alerta.
    - **Tipo –** Nome do tipo de alerta (e-mail ou *sms*).

- **Email -** Guarda o e-mail do usuário.
    - **idEmail -** Código que identifica o e-mail do usuário.
    - **Email –** Guarda o e-mail do usuário.
    - **Usuário_idUsuario –** Código que identifica o usuário.

- **Endereço –** Guarda informações relativas ao endereço do usuário.
    - **idEndereco -** Código que identifica o endereço.
    - **Numero –** Número da residência do usuário.
    - **Complemento –** Complemento da residência**.**
    - **Bairro –** Bairro da residência**.**
    - **Usuário_idUsuario –** Código que identifica o usuário.
    - **Logradouro_idLogradouro –** Código que identifica o logradouro.
    - **Cidade_idCidade –** Código que identifica a cidade.

- **Logradouro –** Contém as informações relativas ao endereço do usuário**.**
    - **idLogradouro -** Código que identifica o logradouro.
    - **Rua -** Nome do endereço do usuário.
    - **CEP –** CEP do endereço**.**

- **Cidade –** Lista as cidades que podem ser escolhidas no sistema.

- **idCidade -** Código que identifica a cidade.
- **Cidade –** Nome da cidade.
- **Estado_idEstado –** Código que identifica o estado.

- **Estado –** Lista os estados que podem ser escolhidos no sistema.
  - **idEstado -** Código que identifica o estado.
  - **Estado -** Nome do estado.

- **Envio –** Contém a informação do status de envio do e-mail ou *SMS*. Pode ser enviado ou não enviado
  - **Data_Envio –** Data de envio da mensagem.
  - **Horário_Envio –** Horário de envio da mensagem.

4.5.1 Interfaces dos Módulos

A interface entre os módulos é mostrada no diagrama a seguir:

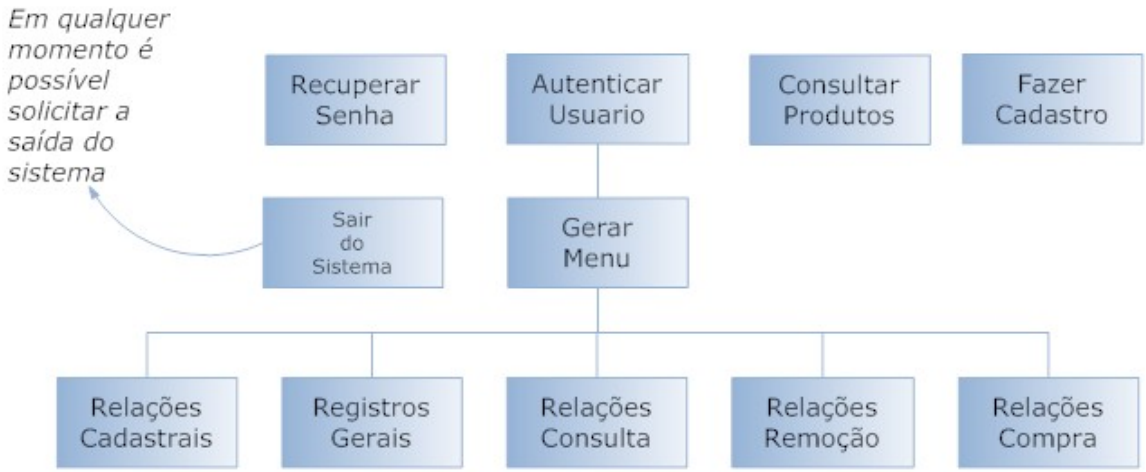


Figura 52 – Interfaces dos módulos

Na tela principal, o usuário terá as seguintes opções:

- Recuperar Senha: caso tenha esquecido sua senha, o usuário poderá clicar nesse link para pedir uma nova senha.

- Consultar Produtos: caso ainda não seja cliente da loja, o usuário poderá clicar nesta opção para consultar os produtos existentes na loja.
- Fazer Cadastro: clicando nesta opção, o usuário poderá se cadastrar na loja para se tornar um cliente.
- Autenticar Usuário: na tela principal existem dois campos para a autenticação do usuário que são o de CPF e senha.

Depois de autenticado no sistema será gerado o respectivo menu caso o usuário seja cliente ou administrador.

É importante ressaltar que uma vez logado no sistema, o usuário poderá sair do mesmo em qualquer momento.

4.5.2 Projeto Detalhado dos Módulos

Neste tópico será detalhado o módulo de cadastrar marca e será mostrado o respectivo código da função para melhor entendimento das relações cadastrais. A primeira parte corresponde a um código *html* da página principal, responsável pela interface com o usuário. Neste página há a chamada para as funções *php* esqueceu senha, cadastro de novo cliente e consulta de produtos. Uma vez autenticado no sistema, será gerado o menu correspondente para o usuário. Caso seja administrador, haverá uma página *html* responsável por chamar todas as funções *php* de administrador. O mesmo ocorre para o cliente. Uma vez que o usuário queira cadastrar alguma marca, o link clicado chama uma função *php* (cadastrar marca) que em seguida realizará a conexão com o banco de dados para inserção dos dados. O acesso ao banco de dados é feito através de comandos *SQL* que estão dentro do código php. Todos os arquivos com a extensão '*.php'* contém uma parte de código *html*, uma vez que este último é responsável pela interface do sistema.

Foram criados arquivos para guardarem constantes pré definidas utilizadas em diversos arquivos. Como exemplo podemos citar as constantes que definem o tamanho da mensagem a ser enviada (número de caracteres), o tamanho do nome e da marca do produto a ser cadastrado, o numero máximo de caracteres aceitos para email. Esta definição facilita a edição do código caso seja necessário aumentar ou diminuir o tamanho dos atributos citados, uma vez que podemos fazer a modificação em apenas um arquivo e não em todos os arquivos onde as constantes são utilizadas. O arquivo foi nomeado de 'constantes.php'.

Todos os outros módulos funcionam de maneira análoga, ou seja, sempre haverá um arquivo html responsável pela interface com o usuário e de acordo com a opção desejada pelo usuário, será chamada a respectiva função *php*. No momento em que as funções *php* são executadas, elas passam a realizar sua respectiva ação, de cadastro, consulta, adição ou remoção comunicando-se com o banco de dados para realizar a operação. Será mostrado abaixo o código do 'index.html' e em seguida o código da função cadastrar marca.

```
<html>
<head>
    <?php html_header(); ?>
</head>
<body>
<?php if ($ret) { echo "    "; mensagens($ret); } ?>
    <form action="login.php" method=post>
    <hr>
    <table border=0 width=100%>
    <tr>
        <td colspan=2 align=center><font size=20><?php echo MSG_TOPO; ?
></font><hr></td>
    </tr>
    <tr>
        <td width=50% align=center><img src="figuras/livros.jpg"></td>
        <td width=50% align=center><table border=0>
        <tr>
            <td colspan=2>Preencha os campos abaixo para fazer o login no
sistema. Se for novo cliente clique no link 'Novo Cliente?'. Caso tenha esquecido sua senha
clique no link 'Esqueceu a senha?'.<br><br></td>
        </tr>
        <table bgcolor="#000000" border="0" cellpadding="1" cellspacing="1">
        <tr><td>
        <table width="300" border="0" cellpadding="5" cellspacing="0">
        <tr>
```

```
                <td bgcolor="#bbbbbb" colspan="2">
                <font face="Arial, Helvetica, sans-serif" size="2" color="#ffffff">
                <b>Identificação do Usuário</b>
                </font>
                </td>
        </tr>
        <tr bgcolor="#ffffff" align="center">
                <td colspan="2">
                <font face="Arial, Helvetica, sans-serif" size="2"
color="#000080"><B>
                </B></font>
                </td>
        </tr>
        <tr bgcolor="#ffffff">
                <td align="right">
                <font face="Arial, Helvetica, sans-serif"
size="2"><B>Identificação</B></font>
                </td>
                <td>
                <INPUT TYPE="text" NAME="cpf" SIZE=15 MAXLENGTH=15>
                </td>
        </tr>
        <tr bgcolor="#ffffff">
                <td align="right">
                <font face="Arial, Helvetica, sans-serif"
size="2"><B>Senha</B></font>
                </td>
                <td>
                        <INPUT TYPE="password" NAME="senha" SIZE=15
MAXLENGTH=15>
                </td>
        </tr>
        <tr bgcolor="#ffffff" align="center">
```

```
                    <td colspan="2">
                    <input type="image" height=30 width=252 border=0 value="Iniciar
Sessão" src="figuras/bot_iniciarsessao.jpg">
                    </td>
            </tr>
            </table>
                    </td>
            </tr>
            </table>
            </FORM>
            <br><br><br>
            <font face="Arial, Helvetica, sans-serif" size="2"><B>
            </b></font>
            </td>
            </tr>
            </table>
            </td></tr>
            </table>
                        <tr>
                    <td colspan=2><a href="procuraprodutoform2.php">Consultar nossos
produtos</a></td>
            </tr>
            <tr>
                    <td colspan=2><a href="novoclienteform.php">Novo Cliente?
</a></td>
            </tr>
            <tr>
                    <td colspan=2><a href="esquecisenhaform.php">Esqueceu a Senha?
</a></td>
            </tr>
            </table></td>
        </tr>
        </table>
```

```
        </form>
    </body>
    </html>
```

Agora segue o código comentado da função cadastrar marca.

```
<?php
        include "/opt/lampp/htdocs/fred_pf/include/funcoes.php"; < -- Inclusão de funções
para a conexão com o BD -- >
        include "/opt/lampp/htdocs/fred_pf/include/constantes.php"; < --Inclusão de macros--
>

        verifica_sessao_inicial_adm(); < -- verifica se o usuário é o administrador -- >

        $ret=$_GET['ret']; < -- variável para retorno de mensagens -- >
?>

<html> < -- inicio do codigo html -- >

<head>
        <?php html_header(); ?>
</head>

<body>
<?php if ($ret) { echo "        "; mensagens($ret); } ?>
        <form action="cadmarca.php" method=post> < -- chama a função cadastrar marca -- >
        <hr>
        <table border=0 width=100%>
        <tr>
                <td colspan=2 align=center><font size=20><?php echo MSG_TOPO; ?
></font><hr></td>
        </tr>
        <tr>
                <td colspan=2 align=right><?php janela_login(); ?></td>
        </tr>
        <tr>
                <td width=50% align=left><img src="figuras/livros.jpg"></td>
                <td width=50% align=center><table border=0>
                <tr>
                        <td colspan=2>Digite o nome da marca que deseja cadastrar no sistema
e clique em cadastrar.<br><br></td>
                </tr>
                <tr>
                        <td>Marca:</td>
                        <td><input type=text name=marca size=<?php echo MARCA_TAM; ?
> maxlength=<?php echo MARCA_MAX; ?></td>
                </tr>
```

```
                <tr>
                        <td><input type=submit value="Cadastrar"><br><br></td>
                </tr>
                <tr>
                        <td colspan=2><a href="menu.php">Menu</a></td>
                </tr>
                </table></td>
        </tr>
        </table>
        </form>
</body>

</html> < -- fim do código html -- >
```

Função cadastrar marca:

```
<?php
        include "/opt/lampp/htdocs/fred_pf/include/funcoes.php";
        include "/opt/lampp/htdocs/fred_pf/include/constantes.php";

        verifica_sessao_inicial_adm();

        $marca = $_POST['marca']; < -- variável para pegar o valor do campo marca -- >

        $ret = inserir_marca($marca); < -- função que está no arquivo funções.php que
realizará a inserção do registro no banco de dados -- >

        if (is_integer($ret)) < -- verifica se o registro foi cadastrado corretamente no banco de
dados -- >
        {
                header("Location:menu.php?ret=".CED_MARCA_CADASTRADA);
                exit;
        }
        else
        {
                header("Location:cadmarcaform.php?ret=$ret");
                exit;
        }
?>
```

5 *Plano de testes*

O objetivo deste tópico é descrever o plano de testes do sistema, de forma que todos os pontos deste sejam percorridos e conseqüentemente testados para verificar o seu bom funcionamento. Serão discutidas formas de se testar o sistema, através de sua divisão em clusters e seus agrupamentos, neste ponto especificaremos que tipo de testes serão realizados.

5.1 Testes Modulares

As interfaces dos módulos devem ser verificadas para ter a garantia de que as informações fluem para dentro e para fora de cada módulo. Um exemplo deste teste seria uma consulta a ser realizada. O usuário entraria com a informação e o sistema trataria a mesma devolvendo as informações desejadas pelo usuário. De maneira geral, pode-se dizer que os testes realizados foram o de fluxo de informações, ou seja, a entrada do usuário, o tratamento dessa informação pelo sistema consultando ou editando o banco de dados e a devolução da informação desejada pelo usuário. A seguir temos os módulos do sistema que foram testados.

O módulo de logoff está ligado a todos os outros módulos. Em qualquer instante o administrador ou o cliente pode fazer a solicitação de logoff do sistema.

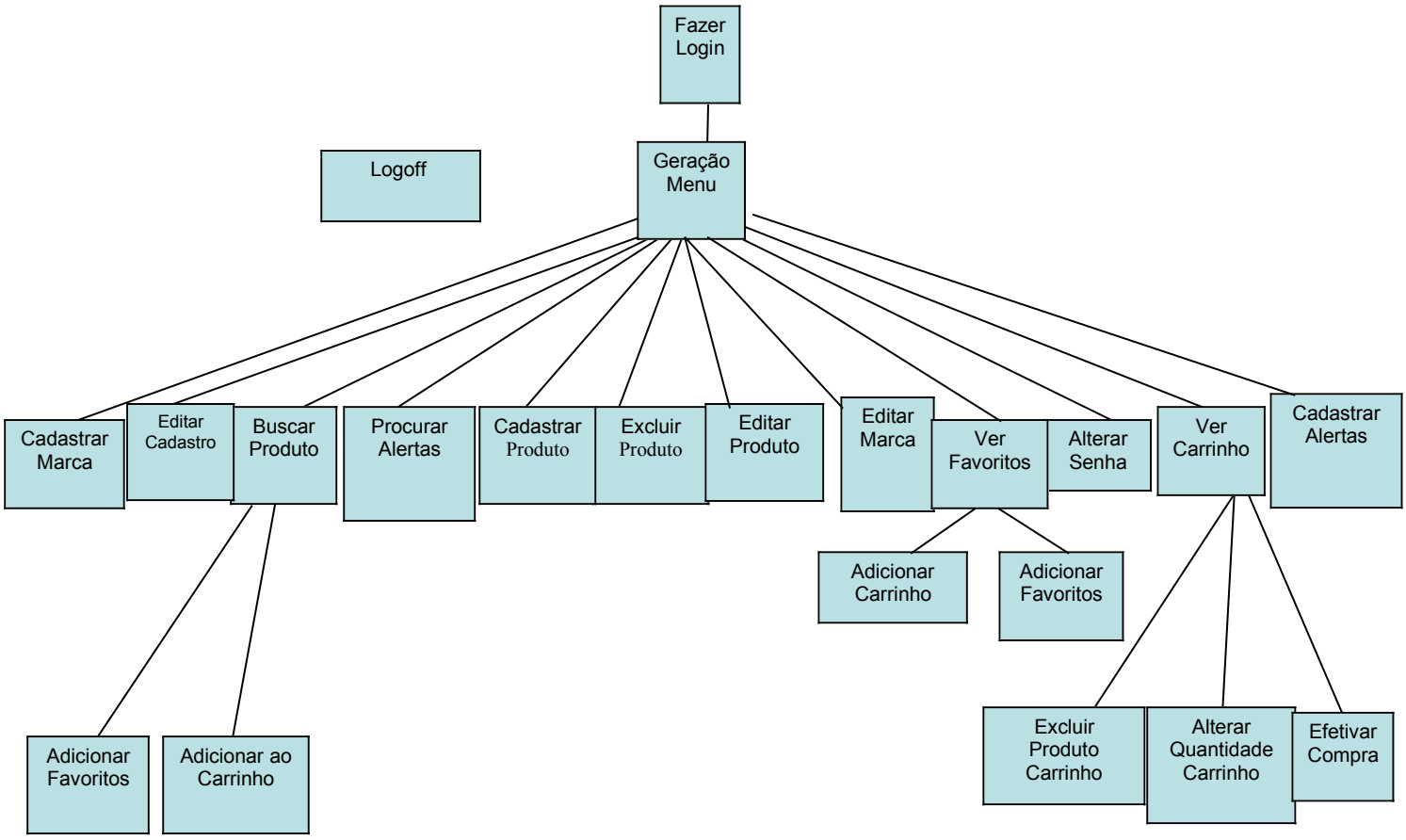


Figura 53 – Testes Modulares

5.2 Testes realizados

Nos testes modulares, foram realizados testes de caixa branca. Este tipo de teste avalia o comportamento interno do componente de software. Essa técnica trabalha diretamente sobre o código-fonte do componente de software para avaliar aspectos tais como: teste de condição, teste de fluxo de dados, teste de ciclos e teste de caminhos lógicos.

5.3 Testes de integração

Na fase de teste de integração o objetivo é encontrar falhas provenientes da integração interna dos componentes de um sistema. Geralmente os tipos de falhas encontradas são de envio e recebimento de dados.

Primeiramente foi analisado a linkagem das páginas *html* com os códigos em *php,* ou seja, foi testado se cada link chama a respectiva função. Em seguida verificou-se o cadastra correto

dos dados no banco. Além disto foram feitas diversas consultas para confirmar se os módulos estavam buscando e enviando os dados corretos para cada função.

5.4 Agrupamento dos módulos em cluster

1. Geração Menu
2. Fazer Login, Logoff Administrador, Logoff Cliente
3. Cadastrar Cliente, Editar Cadastro
4. Buscar Produto, Ver Produto, Adicionar ao Carrinho, Ver Carrinho, Excluir Produto, Editar Produto, Editar Marca, Alterar Quantidade Carrinho, Efetivar Compra
5. Cadastrar Produto, Cadastrar Marca, Excluir Produto
6. Comprar Produto
7. Alterar Senha do Administrador
8. Editar Dados do usuário
9. Cadastrar Alerta

5.4.1 Sequência de integração dos clusters

- 1 – Geração de menus conforme a navegação do usuário
- 2 – Reconhecer o cliente e o administrador.
- 1,2 - o usuário ao se logar na loja deve ser identificado como administrador ou cliente e o sistema deverá identificar essa diferença gerando o menu adequado.
- 1,3 – Gerar o menu correspondente ao cadastro de usuário e de edição de usuário.
- 1,4 – Gerar todos os menus necessários para o cliente buscar o produto, ver o produto, adicionar ao carrinho, ver seu carrinho, excluir produto, alterar a quantidade de produto no carrinho e efetivar a compra.
- 1,5 – Geração de menu para cadastro de produtos, cadastro de marcas e exclusão de produtos e marcas.
- 1,8 – Geração de menu para edição de dados do usuário.
- 1, 9 – Geração de menu para o cadastro de alertas.
- 1,2,3,4,5,6,7,8,9 – Geração de todas as funcionalidades do sistema.

5.4.2 Testes de caixa preta

Em cada passo da seqüência de integração dos clusters, foram realizados testes de caixa preta. São técnicas de teste em que o componente de software a ser testado é abordado como se fosse uma caixa-preta, ou seja, não se considera o comportamento interno do mesmo. Dados de entrada são fornecidos, o teste é executado e o resultado obtido é comparado a um resultado esperado previamente conhecido. Um exemplo foi o teste realizado para a geração do menu correto de acordo com o usuário logado (cliente ou adminitrador).

6 Resultados

Para o servidor do sistema, foi utilizado uma máquina com CPU Pentium IV de 3.0 GHz com 512 MB de RAM. O modem *GSM* foi conectado a esta máquina juntamente com os arquivos *php.* O sistema operacional utilizado foi o Linux Conectiva 10 e o Apache funcionou como servidor.

Foi criado uma pequena base de dados para realizar os testes de amigabilidade e segurança. Foram testados a amigabilidade quanto a interação com o usuário. Dos usuários que testaram a aplicação nenhum mencionou dificuldade de utilização enfatizando a amigabilidade do sistema. A segurança foi testada ao tentar efetuar alguma operação sem estar efetivamente logado. Tentou-se fazer alguns tipos de autenticação inválida, como o de usuários inexistentes e senhas incorretas. Em ambos os casos o sistema respondeu bem, mostrando a mensagem adequada em cão de erro.

A função para compra de livros foi testada com as seguintes possibilidades: compra de produto que está no estoque e compra de produto que não está no estoque. Em ambos os casos a aplicação mostrou a resposta desejada e originou os alertas devidos. Outra funcionalidade testada com esta função foi o envio de alertas, uma vez que o usuário comprando um produto, recebe um *sms* confirmando a compra.

A geração de menu para os respectivos usuários, administrador e usuário, funcionou perfeitamente mostrando os menus corretamente.

Os testes dos alertas foram testados isoladamente. Foram gerados alertas para diversas datas e o sistema respondeu corretamente e os enviou do jeito e nas datas desejadas.

O projeto de desenvolvimento foi estruturado de forma que modificações sejam simples e claras de serem executadas.

Todas as bibliotecas utilizadas e criadas, atenderam plenamente os requisitos de sistema.

7 *Conclusão*

O processo de desenvolvimento deste projeto permitiu a utilização das mais variadas áreas de conhecimento estudadas ao longo do curso de Engenharia Eletrônica e de Computação. Estas abrangeram conhecimentos de Engenharia de Software, banco de dados, programação, telefonia e desenvolvimento para *web*. Este processo foi importante para solidificar os conhecimentos adquiridos na faculdade.

A metodologia utilizada no projeto, a estruturada, foi adequada e proporcionou que desenvolvêssemos o projeto de forma organizada e objetiva. Através dela conseguimos ver os pontos mais críticos do projeto, assim como planejar as melhores soluções. A modelagem relacional e a lista de eventos permitiu visualizar melhor o projeto de maneira que ele fosse desenvolvido da maneira mais adequada.

Todas as ferramentas utilizadas no projeto, atenderam plenamente aos objetivos do sistema como o banco de dados MySQL, a linguagem de programação PHP e o servidor Apache.

Algumas propostas futuras podem ser feitas para a melhoria do sistema como a inserção de imagens dos produtos, além de uma consulta mais rica como citado no modelo lógico onde o usuário poderia escolher diversas funcionalidades do produto e verificar a existência do produto na loja. O sistema poderia ser extendido passando a vender outros produtos como produtos de informática, livros, eletrônicos e etc.

Por se tratar de um software, uma avaliação conclusiva importante é o que diz a metodologia utilizada no projeto (no caso a estruturada). Esta metodologia foi adequada porque garantiu que o projeto fosse desenvolvido de maneira organizada e permitiu analisar os pontos críticos do projeto (Módulo *GSM*).

Portanto, pode-se afirmar que o trabalho mostrado neste projeto foi satisfatório, agregando conhecimento ao desenvolvedor o que será de grande valor ao longo da vida profissional.

Bibliografia

1 - Johnson, Nelson - Advanced graphics in C : programming and techniques, New York : McGraw-Hill Publishing Co., (1987)

2 - Pressman , Roger S - Software Engineering: A Practitioner's Approach, McGraw-Hill Science/Engineering/Math; 6 edition (2004)

3 - Getahead - Direct Web Remoting (DWR), http://getahead.org/dwr. (Acessado em 04 de março de 2007)

4 - Wikipedia – Software testing, http://en.wikipedia.org/wiki/Software_testing. (Acessado em 23 de outubro de 2007)

5 - The Apache Software Foundation – Apache Projects, http://www.apache.org/. (Acessado em 18 de setembro de 2007)

6 - GSMLIB - – GSM Library, http://www.pxh.de/fs/gsmlib/. (Acessado em 18 de setembro de 2007)

Apêndice 1 – Manual do usuário

Esse manual foi feito para atender as dúvidas dos usuários que se caracterizarão pelos clientes e pelo administrador do sistema de compra de produtos via web.

Abaixo estão listadas as tarefas que poderão ser executados de qualquer página e serão vistas nas figurar a seguir.

1.1. Logout

Depois de logado tanto o cliente como o administrador poderão efetuar logout a qualquer momento apenas clicando no menu esquerdo no link Logout.

1.2. Voltar

Todos as páginas possuem a opção voltar para retornar a página principal. Essa opção sempre se encontrará na parte inferior central da página.

1.3. Cliente

Ao acessar a página do sistema o cliente visualizará a página inicial da Loja virtual.

Nesta página o cliente poderá realizar apenas consultar os produtos se ele não estiver cadastrado no sistema, recuperar sua senha através de e-mail ou *sms* ou cadastrar-se no sistema caso ainda não seja.

1.3.1. Cadastro

Ao clicar no link cadastro o cliente será encaminhado para um formulário.

Neste formulário o cliente deverá preencher seus dados pessoais bem como apresentar ao sistema um login de identificação e uma senha. Após o preenchimento do formulário, deve-se enviar os dados para o banco de dados. Isso é feito clicando no botão enviar.

O cadastro também pode não ser efetuado se o login escolhido pelo cliente já constar no sistema ou for inválido, neste caso o sistema avisará com uma mensagem no topo da tela.

1.3.2. Login

Se o cliente já for cadastrado e quiser efetuar a compra de um produto ele deve efetuar o login se identificando para o sistema.

Quando ele fizer isso aparecerá uma página que pedirá seu login e sua senha. Caso ele ainda não tenha se cadastrado ele pode clicar na palavra cadastre-se.

Após efetuado o login aparecerá no menu à esquerda a opção logout no lugar de login, que quando acessada faz com que o cliente se desconecte do sistema não podendo mais realizar nenhuma compra. Aparecerão também mais algumas opções no menu que seráo: ver Carrinho, ver favoritos, consultar produtos e editar dados pessoais. Caso ocorra algum erro na autenticação do cliente o sistema responderá com uma página de erro.

1.3.3. Consulta

Ao clicar no link consultas será apresentado ao cliente um formulário onde ele poderá filtrar os produtos que irão aparecer para ele.

O cliente deve preencher um ou mais campos para executar a filtragem, contudo a informação dada deve ser correta já que o sistema buscará pelo produto que possua todos os parâmetros que foram preenchidos. Por esse motivo se o cliente não tiver certeza sobre alguns dos campos, é preferível que esse campo fique em branco deixando, portanto, o sistema livre naquele parâmetro.

1.3.4. Carrinho

Quando o cliente quiser ver o que adicionou ao carrinho ou ainda finalizar sua compra deve ir ao menu direito e clicar na palavra Ver Carrinho que só aparecerá se ele estiver logado.

Nesta página o cliente pode verificar os produtos que já colocou no carrinho, modificar a quantidade de cada produto que gostaria de adquirir e pode ainda retirar o item do carrinho caso tenha mudado de idéia.

1.3.5. Esqueceu senha

Caso o cliente tenha esquecido a senha, ele poderá clicar no link esqueci_senha para pedir uma nova. O sistema oferece duas maneiras de envio da senha, ou através de e-mail ou através de *sms*. Ao clicar no link de esqueci_senha, o cliente terá que se identificar com o seu CPF e escolher o meio de envio da senha.

1.4. Administrador

O administrador do sistema deverá incluir, excluir e editar produtos do acervo, receberá um aviso do sistema quando houver baixa quantidade de determinado produto no estoque. Para efetuar essas tarefas ele deve se logar no sistema como root.

1.4.1. Cadastrar Produto

Para incluir um novo produto o administrador deve clicar no menu à direita no item cadastrar novo produto. Nesta página o administrador deverá preencher os campos para incluir os dados do novo produto, incluindo a quantidade. Caso não coloque a quantidade de produtos, o sistema preencherá com zero por padrão.

1.4.2. Editar produto

Para fazer a alteração de algum produto o administrador poderá ir à opção editar produto. Aparecerá, então, o formulário de busca e o resultado da busca assim como o do cliente, contudo, quando ele clicar no produto diferentemente do cliente irá aparecer um formulário como o de inserir produto e ele poderá fazer as alterações necessárias.

1.4.3. Excluir Produto

O procedimento de exclusão de produto é bem semelhante ao de edição, clica-se na opção Excluir Produto, e assim como em editar aparecerá o formulário de busca e a página de resultados, só que desta vez ao clicar no produto o administrador estará excluindo o produto.

1.4.4. Cadastrar Marca

O produto para ser cadastrado no sistema deverá estar relacionado com uma marca. Esta marca é cadastrada pelo administrador assim como o produto. Aparecerá uma tela onde o usuário irá cadastrar a marca.

1.4.5. Editar Marca

Assim como o produto o administrador poderá editar a marca selecionando o link Editar_Marca no menu principal da administrador.

1.4.6. Excluir Marca

Na tela principal de administrador, o próprio poderá excluir a marca clicando no link Excluir_Marca.

1.4.7. Envio de promoções e novos produtos

A través do link cadastra promoções, o administrador poderá enviar ao cliente os produtos recém cadastrados com seus respectivos preços. O cliente receberá as informações tanto em seu e-mail quanto em seu celualar.