

**LIGA DE ENSINO DO RIO GRANDE DO NORTE
FACULDADE NATALENSE PARA O DESENVOLVIMENTO
DO RIO GRANDE DO NORTE
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO**

**IMPLEMENTAÇÃO DE UM COMÉRCIO ELETRÔNICO PARA EMPRESAS DO
SETOR ALIMENTÍCIO QUE TRABALHAM COM ENTREGA EM DOMICÍLIO**

Rodrigo de Souza Pires

**NATAL/RN
2002**

RODRIGO DE SOUZA PIRES

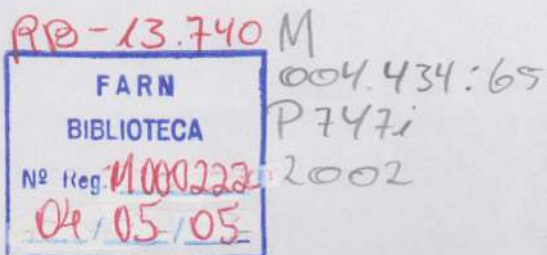
IMPLEMENTAÇÃO DE UM COMÉRCIO ELETRÔNICO PARA EMPRESAS DO SETOR ALIMENTÍCIO QUE TRABALHAM COM ENTREGA EM DOMICÍLIO

Monografia apresentada ao curso de Bacharelado em Sistemas de Informação da Faculdade Natalense para o Desenvolvimento do Rio Grande do Norte como requisito parcial para a obtenção do grau de Bacharel em Sistemas de Informação.

Orientador:

Fellipe Araújo Aleixo, MsC.

**NATAL/RN
2002**



Pires, Rodrigo de Souza

Implementação de um Comércio Eletrônico para Empresas do Setor Alimentício que Trabalham com Entrega em Domicílio - Rodrigo de Souza Pires. Dezembro, Natal: FARN, 2002.

66p.

Monografia (Graduação em Bacharelado em Sistemas de Informação) - Faculdade Natalense para o Desenvolvimento do Rio Grande do Norte.

1. Linguagem para aplicações comerciais – Monografia.
2. Comércio eletrônico – Monografia I Título

RN/FARN/BC

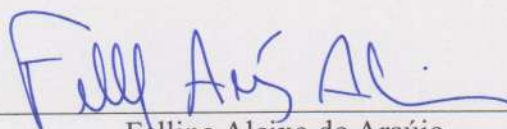
CDU 004.434:65

LIGA DE ENSINO DO RIO GRANDE DO NORTE
FACULDADE NATALENSE PARA O DESENVOLVIMENTO
DO RIO GRANDE DO NORTE
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

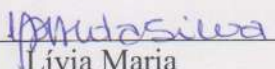
IMPLEMENTAÇÃO DE UM COMÉRCIO ELETRÔNICO PARA EMPRESAS DO SETOR
ALIMENTÍCIO QUE TRABALHAM COM ENTREGA EM DOMICÍLIO

RODRIGO DE SOUZA PIRES

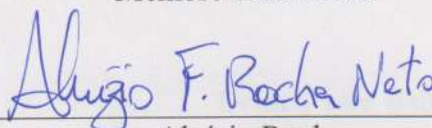
Monografia apresentada e aprovada em 14 de Dezembro de 2002 pela banca examinadora
composta pelos seguintes membros:



Felipe Aleixo de Araújo
Orientador



Livia Maria
Membro examinador



Aluizio Rocha
Membro examinador

Natal, 14 de Dezembro de 2002

AGRADECIMENTOS

Agradeço à Deus, por ter me dado essa chance de estar aqui desfrutando desse momento único.

Agradeço aos meus pais, irmãos e a todos os meus familiares, que ajudaram, mesmo que oculta e silenciosamente, para mais esse passo na evolução da minha vida acadêmica.

Agradeço à minha amiga, companheira, amante e finalmente companheira, pela participação ativa na realização deste trabalho e por ter me ajudado até agora.

Agradeço ao meu orientador, por ter depositado sua confiança em mim para a execução desse projeto.

Agradeço à minha coordenadora, por ter compreendido as minhas dificuldades e por estar sempre disponível em qualquer situação.

Agradeço aos meus amigos que, naquela volta, ajudaram a continuar me motivando.

Agradeço aos meus companheiros (meus irmãos) que sempre me ajudaram em todos os aspectos, tanto emocionalmente quanto intelectualmente, como também, não poderei deixar, nas palavras de vida.

Agradeço a todos que, de alguma maneira, contribuíram para a realização deste trabalho.

Dedico este trabalho aos meus familiares e, em especial, aos meus pais, Airton de Carvalho Pires e Marielma Marta Souza Pires, pela compreensão e confiança que em mim depositaram durante toda a sua realização.

AGRADECIMENTOS

ÍNDICE

Agradeço à Deus, por ter me dado essa chance de estar aqui desfrutando desse momento único.

Agradeço aos meus pais, irmãs e a todos os meus familiares, que ajudaram, mesmo que ocultamente, para mais esse passo na evolução da minha vida acadêmica.

Agradeço à minha amiga, companheira, amante e finalmente namorada, pela participação ativa na realização deste trabalho e por ter me agüentado até agora.

Agradeço ao meu orientador, por ter depositado sua confiança em mim para a execução desse projeto.

Agradeço à minha coordenadora, por ser compreensiva nos momentos difíceis e por estar sempre disponível em qualquer situação.

Agradeço aos meus amigos que, naquela velha batalha, continuaram me incentivando.

Agradeço aos meus companheiros (meus *brothers*), que sempre me ajudaram em todos os aspectos, tanto emocionalmente quanto bibliograficamente, como também, não poderia faltar, nas cervejadas da vida.

Agradeço à todos que, de alguma maneira, contribuíram para a realização deste curso.

2.2.3	SQL	24
3	Implementação de um Protótipo voltado para Controle Eletrônico	27
3.1	Diagrama de Classes	28
3.2	Diagrama de Utilização	30
3.2.1	Endereço	30
3.2.2	Ativa a Sonda	41
3.2.3	Ver Endipiv	40
3.2.4	Gerenciar Início do Contêiner	53
3.3	Fluxo de Dados	36
4	Conclusões e Perspectivas	38
REFERÊNCIAS BIBLIOGRÁFICAS		60
APÊNDICE A - CÓDIGO HTML		62
APÊNDICE B - CÓDIGO PHP		64
APÊNDICE C - CÓDIGO JAVA		66

ÍNDICE

RESUMO.....	8
ABSTRACT.....	9
1. Introdução	10
1.1 Apresentação	10
1.2 Objetivos	10
1.3 Motivação	11
1.4 Metodologia	12
1.5 Organização do Trabalho	13
2. Referencial Teórico.....	14
2.1 Redes de Computadores.....	14
2.2 Linguagens.....	15
2.2.1 HTML	15
2.2.2 XML.....	20
2.2.3 JAVA	21
2.2.4 JSP.....	22
2.2.5 SQL	24
3. Implementação de um Protótipo voltado para Comércio Eletrônico.....	27
3.1 Diagrama de Classe.....	28
3.2 Cenários de Utilização	30
3.2.1 Cadastro	30
3.2.2 Alterar Senha	41
3.2.3 Ver Cardápio.....	49
3.2.4 Gerenciar Itens do Cardápio	55
3.3 Banco de Dados	56
4. Conclusões e Perspectivas	58
REFERÊNCIAS BIBLIOGRÁFICAS.....	60
APÊNDICE A – CÓDIGO HTML	62
APÊNDICE B – CÓDIGO JSP.....	64
APÊNDICE C – CÓDIGO JAVA	65
Figura 31 – CheckCep.jsp – Requisição CPF	32
Figura 32 – Diagrama de Colaboração CheckCep	33
Figura 33 – AlteraCadastro.jsp – Requisição Novo Endereço	39
Figura 34 – AlteraCadastro.jsp – Informa Novo Endereço	39
Figura 35 – Diagrama de Colaboração AlteraCadastro	40
Figura 36 – Diagrama de Atividades AlteraCadastro, Servlet CheckCep, ServletCheckCep e ServletCadastro	42
Figura 37 – Alterar Senha	41
Figura 38 – checkCep.html – Requisição Identificação	42
Figura 39 – Diagrama de Colaboração CheckCep	42
Figura 40 – AlteraSenha.jsp – Requisição Nova Senha	43
Figura 41 – AlteraSenha.jsp – Sucesso Alteração Senha	43
Figura 42 – Diagrama de Colaboração AlteraSenha	44

LISTA DE FIGURAS

Figura 1 – Protocolos e Redes no Modelo TCP/IP	10
Figura 2 – Exemplo de HTML.....	17
Figura 3 – código do formulário do tipo TEXT.....	18
Figura 4 – Resultado	18
Figura 5 – código do formulário do tipo PASSWORD	18
Figura 6 – Resultado	18
Figura 7 – código do formulário do tipo CHECKBOX.....	18
Figura 8 – Resultado	18
Figura 9 – código do formulário do tipo RADIO	19
Figura 10 – Resultado	19
Figura 11 – código do formulário do tipo SUBMIT.....	19
Figura 12 – Resultado	19
Figura 13 – código do formulário do tipo SELECT	20
Figura 14 – Resultado	20
Figura 15 – Aplicação XML	21
Figura 17 – Aplicação JSP.....	24
Figura 18 – Diagrama de Classes.....	29
Figura 19 – Index.html – pagina inicial do sistema	30
Figura 20 – Cadastro.html – Opções para acesso ao cadastro	31
Figura 21 – Identific.html – Requisição Identificação.....	32
Figura 22- Diagrama de Colaboração ChecaLogin	32
Figura 23 – checaCep.jsp – Requisição CEP.....	33
Figura 24 – Diagrama de Colaboração ChecaCEP	33
Figura 25 – cadastra.jsp – Requisição Complemento	34
Figura 26 – cadastra.jsp – Requisição Senha.....	35
Figura 27 – Diagrama de Colaboração Cadastra	35
Figura 28 – Diagrama de Atividades Inserir Cadastro: ServletChecaLogin, ServletChecaCep e ServletCadastra.....	36
Figura 29 – ChecaLogin.html – Requisição Identificação.....	37
Figura 30 – Diagrama de Colaboração ChecaLogin.....	37
Figura 31 – ChecaCep.jsp – Requisição CEP	38
Figura 32 – Diagrama de Colaboração ChecaCep	38
Figura 33 – alteraCadastro.jsp – Requisição Novo Endereço.....	39
Figura 34 – alteraCadastro.jsp – Informa Novo Endereço	39
Figura 35 – Diagrama de Colaboração AlteraCadastro	40
Figura 36 – Diagrama de Atividades Alterar Cadastro: ServletChecaLogin, ServletChecaCep e ServletCadastra.....	40
Figura 37 – Alterar Senha	41
Figura 38 – checaLogin.html - Requisição Identificação	42
Figura 39 – Diagrama de Colaboracao checaSenha	42
Figura 40 – alteraSenha.jsp - Requisição Nova Senha	43
Figura 41 – alteraSenha.jsp – Sucesso Alteração Senha.....	43
Figura 42 – Diagrama de Colaboração AlteraSenha.....	44

Figura 43 – Diagrama de Atividades Alterar Senha: ServletChecaLogin e ServletNovaSenha.....	44
Figura 44 – checaLogin.html – Requisição Identificação.....	45
Figura 45 – Diagrama de Colaboração ChecaLogin.....	45
Figura 46 – Requisição Frase.....	46
Figura 47 – checaDados.jsp – Confirmação de Dados.....	46
Figura 48 – Diagrama de Colaboração Checa Dados.....	47
Figura 49 – senhaEsque.jsp – Requisição Nova Senha.....	47
Figura 50 – Diagrama de Colaboração – Requisição Nova Senha.....	48
Figura 51 – Diagrama de Atividades Alterar Senha Esquecida: ServletChecaLogin, ServletChecaDados e ServletSenhaEsque.....	48
Figura 52 – Cardapio.xml – Cardápio.....	49
Figura 53 – Cardapio.xml – Cardápio.....	50
Figura 54 – cesta.jsp – Cesta de Pedido.....	51
Figura 55 – Diagrama de Colaboração Inserir Item Cardápio.....	51
Figura 56 – Diagrama de Colaboração Excluir Item Cardápio.....	51
Figura 57 – Diagrama de Atividades: ServletCesta.....	52
Figura 58 – finaliza.jsp – Finalizar Pedido.....	53
Figura 59 – Diagrama de Colaboração Finalizar Pedido.....	53
Figura 60 – finaliza.jsp – Finaliza Pedido.....	54
Figura 61 – Diagrama de Atividades: ServletFinaliza.....	54
Figura 62 – Diagrama de Entidades e Relacionamentos.....	55

RESUMO

Este trabalho apresenta uma solução de desenvolvimento de um sistema de comércio eletrônico, voltado para as empresas do setor alimentício. Para estes fins, utiliza-se arquitetura de redes de computadores e Internet, baseadas no protocolo TCP/IP. O trabalho, denominado "Especificação de um Comércio Eletrônico para Empresas Alimentícias que Trabalham com Entrega em Domicílio: Implementação de uma Infra-Estrutura Mínima", possibilita a exploração dos serviços de *e-business* (negócios eletrônicos) pelas empresas locais de *fast-food*, aumentando a qualidade no atendimento aos seus consumidores, bem como a redução de custos e complexidade dos sistemas atuais.

Este trabalho apresenta uma descrição da solução proposta para desenvolvimento, descrição da interface e o banco de dados utilizado. As decisões de implementação do projeto, visão geral do protótipo desenvolvido e contribuições da ferramenta são também apresentadas, em conjunto com as conclusões e perspectivas para trabalhos futuros.

ABSTRACT

This work presents a solution for development of a e-commerce system, gone back to the companies of the nutritious. For these ends, it is used the computers networks architecture and Internet, based on the protocols TCP/IP. The work, denominated "Specification of a e-commerce for Nutritious Companies that Work with Delivery in Home: Minimum Infrastructure Implementation", makes possible the exploration of the services of e-business (electronic businesses) for the local companies of fast-food, increasing the quality in the attendance to your consumers, as well as the reduction of costs and complexity of the current systems.

In your continuity, it comes a description of the solution proposed for your development, through a general vision of the proposed system and your operation, as well as of a description of the chosen interface and of the relational database used. The project's implementation decisions, general vision of the developed prototype and contributions of the tool are also presented, together with the conclusions and perspectives for future works.

Objetivo e linguagem inter-ativos. São duas a Internet a melhor forma de transporte mensagens (dados) de uma máquina para outra, independentemente de onde as máquinas estejam na mesma rede ou em redes interconectadas. O TCP é um protocolo de transporte responsável por garantir a entrega dos dados para o destino correto. O IP é responsável por garantir a entrega dos dados para o destino correto. O IP não oferece qualquer garantia que os dados sejam entregues na forma apropriada. Como o TCP adiciona os serviços de entrega e recuperação de dados que são necessários. Os dados podem também chegar fora de ordem, quando o tempo de entrega ao TCP é superior ao da rede de destino.

Depois do nascimento do TCP/IP, os protocolos e serviços de rede foram desenvolvidos para que pudéssemos ter disponíveis os melhores serviços que a Internet oferece, como o serviço de correio eletrônico (SMTP), transferência de arquivos (FTP), serviços de informação e transferência, entre outros.

	TELNET	FTP	SMTP	DNS	Aplicação
Protocolos	TCP		UDP		Transporte
	IP				Rede
Rede	ARPANET	SATNET	Packet Radio	LAN	Redes de Dados

Figura 1 – Protocolos e Redes no Modelo TCP/IP

1. Introdução

1.1 Apresentação

A tecnologia de redes de computadores foi criada em função da necessidade de interligação e comunicação entre os vários computadores, outras redes e bases de pesquisa existentes que se encontravam geograficamente distribuídas.

A Internet pode ser vista como um conjunto de sub-redes ou sistemas autônomos conectados entre si. Não existe uma estrutura pré-definida, mas diversas redes interligadas. Fazem parte da Internet redes regionais (nível médio) e redes locais de muitas universidades, empresas e provedores de serviços (Tanenbaum, 1997).

Com o tempo, cresceu a necessidade de criação de convenções utilizadas para a interconexão de computadores. Convenções estas chamadas de protocolos.

O elemento que mantém a padronização da Internet é o protocolo IP (*Internet Protocol*), que foi projetado tendo como objetivo a ligação inter-redes. Sua tarefa é fornecer a melhor forma de transportar mensagens (datagramas IP) desde a origem até o destino, independente de estas máquinas estarem na mesma rede ou em redes intermediárias. O TCP é um protocolo de transporte orientado à conexão e foi projetado para oferecer uma comunicação fim a fim confiável, em uma rede não-confiável. O IP não oferece qualquer garantia que os datagramas serão entregues da forma apropriada. Cabe ao TCP administrar os temporizadores e retransmitir sempre que necessário. Os datagramas também podem chegar fora de ordem, cabendo também ao TCP reorganizá-los na sequência correta.

Depois do nascimento do TCP/IP foi necessária a criação de outros protocolos para que pudessem ser disponibilizados os milhares serviços que a Internet oferece, como o serviço de correio eletrônico (SMTP), transferência de arquivos (FTP), serviços de informação e transferência, entre outros.

					Camadas
Protocolos	TELNET	FTP	SMTP	DNS	Aplicação
	TCP		UDP		Transporte
	IP				Rede
Redes	ARPANET	SATNET	Packet Radio	LAN	Física + Enlace de Dados

Figura 1 – Protocolos e Redes no Modelo TCP/IP

Com a disponibilização destes serviços, as empresas aspiraram uma nova estrutura no seu setor administrativo, tendo que arriscar mais, buscando sempre um diferencial dentre as demais no seu ramo de trabalho.

A Internet tem aberto novos horizontes para as empresas, possibilitando o desenvolvimento de sistemas que permitem a integração das empresas com seus clientes (Furgeri, 2001).

Foi então que surgiu o *e-business* (negócio eletrônico), que pode ser estruturado em cinco níveis de maturidade abrangendo o *marketing* digital: vendas, pagamentos, atendimento, entrega de produtos e suporte e serviços pós-venda.

O *e-business* abriu as portas para a incorporação do comércio eletrônico na Internet, que tem interessado as pessoas, tanto físicas como jurídicas, no que diz respeito a essa forma de se fazer negócios.

Por definição, o comércio eletrônico é toda a atividade de compra e venda realizada com o auxílio de recursos eletrônicos. O comércio eletrônico é, na verdade, praticado há anos, porém só foi denominado como *e-commerce* a pouco tempo. Sempre que faz-se uma compra utilizando cartão de crédito, pagamentos com cheques eletrônicos, vendas concretizadas por fax ou negociadas por *e-mail*, está se caracterizando o comércio eletrônico (Nakamura, 2001).

Hoje, entretanto, foram incorporados ao seu conceito novos recursos tecnológicos, provenientes da área de informática e redes de computadores, para possibilitar a realização das operações tradicionais de compra, venda, transferência, dentre outras, com mais rapidez, segurança, eficiência e custo relativamente menor.

A atividade de venda pode ser dividida em três pilares: pré-venda, pós-venda e a venda efetiva (transação). A estratégia do *e-commerce* pode ser baseada em um ou mais pilares, ou ainda ser implementada parte por parte.

1.2 Objetivos

Neste projeto apresenta-se, a partir de uma análise orientada a objetos realizada em um trabalho correlato (Santos, 2002), uma proposta de implementação de um protótipo de um comércio eletrônico em empresas do setor alimentício, mais precisamente empresas que trabalham na comercialização de *fast food*, realizando entrega à domicílio na grande Natal.

Portanto, este trabalho tem como principal objetivo desenvolver um protótipo de um sistema, voltado à área de comércio eletrônico, que tenha como escopo principal o atendimento às necessidades comerciais de empresas que trabalham no ramo alimentício estilo *fast food*.

1.3 Motivação

A rapidez e a eficiência no atendimento de televendas são pontos cruciais para o consumidor que procura por empresas que oferecem este tipo de serviço. Tendo isto como ponto de partida abordaremos alguns problemas inerentes a este serviço.

Para inserir o serviço de televendas, o estabelecimento deve dispor de um funcionário destinado para o teleatendimento. Dependendo da solicitação do estabelecimento comercial, poderão ocorrer congestionamentos na linha telefônica, causando transtornos e perda de clientes para o estabelecimento em questão. Para tentar contornar este problema, se poderia pensar na aquisição de uma outra linha telefônica e na contratação, subsequente, de um atendente para esta nova linha, o que levaria ao aumento dos custos, estes repassados ao consumidor. Muitas vezes o consumidor não tem acesso ao cardápio através de meios de publicidade, tais como: *folders*, catálogos, televisão. A falta do conhecimento dos detalhes a cerca de cada produto, leva a demora na finalização do pedido e a não compreensão das partes envolvidas no pedido (consumidor - atendente).

Como o foco deste trabalho é a cidade do Natal, foi identificado que não existem muitas empresas do setor alimentício que incorporem aos seus serviços o atendimento via Internet, apesar de que a tendência da maioria dos consumidores é dispor de acesso à rede. E ainda quando oferecido, o serviço de *e-commerce* não é plenamente satisfatório, mas arcaico e com algumas limitações, dentre elas, a pouca divulgação e a falta de padronização.

A implantação do comércio eletrônico no estabelecimento comercial implicará, para a empresa, redução de gastos com anúncios publicitários, visto que o comércio eletrônico é disponível a qualquer dia do ano e a qualquer hora do dia ou da noite. Com isto, o estabelecimento reduz custos e se posiciona privilegiadamente à frente daqueles que não possuem este tipo de recurso.

E, para o consumidor, pode-se vislumbrar os seguintes benefícios:

- Acesso ao cardápio com a utilização de recursos gráficos;
- Otimização do tempo de pedido;
- Criação e utilização de cestas de pedidos, com a função de armazenar o(s) item(s) escolhido(s) pelo cliente, realizando uma simulação ou a efetivação deste;
- Alteração e / ou cancelamento do pedido antes do seu pagamento;
- Elaboração de perfil de usuário ou pedido personalizado de acordo com a constância de um determinado cliente quanto aos seus pedidos.

Espera-se que esta tecnologia incorpore praticidade, versatilidade e rapidez no atendimento aos clientes da grande Natal, aproximando e aperfeiçoando as relações prestadora de serviços / consumidor.

Então, a intenção deste trabalho é a de propor uma padronização para sistemas de atendimento ao cliente via *web*, de fácil acesso e de baixo custo.

1.4 Metodologia

A metodologia utilizada para o desenvolvimento deste trabalho pode ser resumida pelos seguintes passos:

- Pesquisa bibliográfica relacionada aos assuntos envolvidos neste trabalho, dentre os quais relacionados as redes de computadores e seus protocolos básicos de comunicação, Internet, televendas e comércio eletrônico. Desta maneira, foi possível traçar o estado-da-arte destas tecnologias e posicionar claramente o trabalho apresentado com relação a outras abordagens existentes.
- Realização de um estudo aprofundado das tecnologias voltadas para desenvolvimento com utilização de recursos de Banco de Dados para dar um suporte teórico às decisões de projeto adotadas, bem como para a definição dos serviços a serem oferecidos pelo ambiente proposto (a especificação das funcionalidades da

ferramenta foi feita com base nas informações obtidas através de especialistas do setor alimentício);

- Modelagem de um banco de dados relacional para o problema, execução de testes no banco de dados e realização das adaptações necessárias à implementação;
- Desenvolvimento de um protótipo do ambiente baseado na análise prévia e realização de testes de validação de seu funcionamento;
- Teste e correção de falhas iniciais de projeto, assim como inclusão de novas funcionalidades ao ambiente;
- Avaliação dos resultados obtidos com os testes de validação e usabilidade do protótipo.

1.5 Organização do Trabalho

Este trabalho é dividido em quatro outros capítulos além desta apresentação inicial, descritos brevemente a seguir.

O segundo capítulo, apresenta os conceitos de redes de computadores, Internet, protocolos de comunicação, linguagens de programação voltadas para Internet e banco de dados relacional. O capítulo segue com uma descrição de comércio eletrônico e finaliza com o posicionamento deste trabalho dentro da área de interesse relacionada.

O terceiro capítulo, apresenta a descrição da solução proposta para desenvolvimento de um sistema de comércio eletrônico, voltado para o setor alimentício, implementado neste trabalho. Inicia-se com uma visão geral do sistema proposto e seu funcionamento, sendo em seguida apresentada uma descrição da interface escolhida. O capítulo ainda apresenta o banco de dados relacional e uma análise das alternativas adotadas durante o projeto, e finaliza com a apresentação do protótipo.

Finalizando este documento, o quarto capítulo, apresenta a conclusão do trabalho, apontando as perspectivas de trabalhos futuros.

2. Referencial Teórico

Este Capítulo faz uma breve introdução dos conceitos relevantes para o entendimento deste trabalho. Inicialmente, apresenta-se a tecnologia de redes de computadores, suas aplicações e tentativas de padronização. Em seguida, as linguagens de programação utilizadas para desenvolvimento de sistemas voltados para a Internet são rapidamente descritas, como também conceitos relevantes de banco de dados relacional são apresentados.

2.1 Redes de Computadores

As redes de computadores podem ser usadas para inúmeros serviços, tanto por pessoas físicas quanto por pessoas jurídicas. Para as empresas, as redes de computadores pessoais que utilizam servidores compartilhados com frequência oferecem flexibilidade e uma boa relação custo/desempenho. Para os indivíduos, as redes oferecem acesso a uma série de serviços, informações e entretenimento.

Algumas padronizações foram desenvolvidas e adotadas para que a tecnologia de redes de computadores pudesse ser disseminada. Dentre elas estão o serviço de correio eletrônico, de transferência de arquivos e serviço de informação.

O serviço de correio eletrônico tem como finalidade a comunicação assíncrona entre os computadores da rede. Dentro da Internet, o serviço de correio eletrônico é disponibilizado via protocolo SMTP (*Simple Mail Transfer Protocol*). As mensagens de correio eletrônico são entregues quando a máquina de origem estabelece uma conexão TCP com a porta 25 da máquina de destino. Um cliente SMTP permanece escutando essa porta e aceita as conexões recebidas, copiando as mensagens nelas contidas para as caixas de correio apropriadas (Tanenbaum, 1997).

Em uma rede TCP/IP, um outro protocolo que presta serviço é o FTP (*File Transfer Protocol*), que possibilita a transferência confiável de arquivos através da rede de computadores, utilizando para tanto duas portas conhecidas, a porta 20 para o controle da conexão e a porta 21 para a transferência dos dados (Naugle, 1994).

O protocolo de transferência padrão da WWW (*World Wide Web*), que é a estrutura arquitetônica que permite o acesso a documentos vinculados e espalhados por milhares de máquinas na Internet, é o HTTP (*Hipertext Transfer Protocol*). Todos os sites da *Web* têm processos servidores que escutam a porta 80 TCP aguardando

conexões dos clientes (geralmente *browsers* ou navegadores). Depois de estabelecida a conexão, o cliente envia uma solicitação e o servidor envia uma resposta. A conexão é, então, liberada. O HTTP é quem define as solicitações e respostas válidas. Em outras palavras, o protocolo HTTP consiste em dois itens razoavelmente distintos: um conjunto de solicitações dos *browsers* aos servidores e um conjunto de respostas que retornam no caminho inverso.

2.2 Linguagens

Para a implementação deste trabalho foi explorado o conceito de arquitetura 3-camadas. A expressão 3-camadas refere-se à quantidade de camadas em que o software está logicamente dividido, significando que a interface do aplicativo está nos clientes, o acesso e a atualização dos dados ocorre nos servidores de dados e a lógica da aplicação está separada desses dois elementos, nos servidores de aplicação.

2.2.1 HTML

A *Hypertext Markup Language* (Linguagem de Marcação de Hipertexto - HTML) é um acrônimo que combina dois importantes conceitos: o conceito de hipertexto e de uma linguagem de marcação. Com HTML é possível apresentar e organizar as informações na Internet. O propósito do hipertexto é facilitar a navegação não-linear entre documentos. Um documento de hipertexto contém links ativos, ou ponteiros, para outras fontes de informação. Isto significa que com alguns clicks você pode navegar entre diferentes fontes de informação. É fato que HTML e a Internet se relacionam e são baseados na relação cliente/servidor, permitindo a troca de informações ponto a ponto em qualquer parte do mundo (HTML, 1999).

No caso do hipertexto, a fonte de informação é texto. Hipermídia é uma extensão do hipertexto, e inclui acesso a outras fontes de informação, incluindo imagens, vídeos, som e multimídia que podem ser incorporadas a documentos HTML.

Tanto texto e imagens podem ser usados em uma página web como *links* para outras páginas web e recursos na Internet. Clicando em um destes *hiperlinks* outra página web será carregada em seu *browser*, ou será feito o *download* do tipo de mídia associado com o *link*.

A Grande Teia (*World Wide Web*), tem sido responsável pelo rápido crescimento e aceitação do hipertexto como ferramenta de navegação e exploração de informação, por sua natureza intuitiva.

Cada *Hiperlink* pode levar: a outra página *web* com mais *hyperlinks*, a uma imagem, som, ou animação que irá abrir outra aplicação, a um programa executável, a um arquivo qualquer, a outra localização na mesma página *Web*.

O principal motivo pelo qual usaremos esta tecnologia é que não importa se você utiliza um *browser Windows*, *Macintosh* ou *Unix*. Enquanto você estiver utilizando um *browser* compatível com HTML, você pode visualizar documentos HTML armazenados em um servidor. Isto significa que um documento HTML pode ser visualizado utilizando um *browser* em qualquer uma destas plataformas e a página será exibida quase que de forma idêntica. Esta é a razão pela qual HTML tem crescido e vem se tornando o padrão primário para a criação e visualização de documentos de hipertexto e hiperídia na Grande Teia Mundial.

A HTML permite que caracteres especiais sejam representados por seqüências de escape, indicadas por três partes: um & inicial, um número ou cadeia de caracteres correspondente ao caracter desejado, e um ; final.

Na Figura 2, para que os caracteres "<" e ">" fossem exibidos, foi necessária a utilização de códigos especiais. As representações dos tags <P> e
 foram escrita na forma de código (<P> e
, respectivamente). Do contrário, essas representações não seriam mostradas na página, mas interpretadas com os efeitos reais de cada um dos tags.

```
<HTML>
<HEAD>
<TITLE>Primeiro exemplo</TITLE>
</HEAD>
<BODY>
<H1>Título principal</H1>
Este é o texto do primeiro exemplo.
Para dividir os parágrafos, usa-se o tag &lt;P&gt;.<P>
Este é o segundo parágrafo. Para colocar um novo título depois
de um parágrafo não é necessário colocar o tag &lt;P&gt;..
<H2>Título secundário</H2>
Para abrir uma linha, usa-se o tag &lt;br&gt;,<br>
certo?
<H3>Uma lista</H3>
<UL>
<LI>Item 1
<LI>Item 2
<LI>Item 3
</UL>
</BODY>
</HTML>
```

Figura 2 – Exemplo de HTML

Assim que o exemplo for salvo no editor de texto com a extensão “.HTM” ou “.HTML”, pode-se visualizá-lo em um navegador.

O primeiro exemplo mostra vários trechos de texto marcados por códigos colocados entre os caracteres “<” e “>”. Esses códigos, chamados de *tags*, são responsáveis pela marcação do texto em função de seu papel dentro do documento. O título principal é marcado com os tags <H1> e </H1>, enquanto os parágrafos são separados pelo tag <P>.

Existem dois tipos de tags: alguns são formados aos pares, indicando o início e o fim do trecho afetado, como o par <H1> e </H1>; outros podem ser colocados individualmente, como o <P>, que simplesmente insere um espaço para dividir parágrafos. Mais adiante, mostraremos que os tags também podem receber atributos.

Existem quatro pares de tags que devem ser sempre colocados na página. O par de tags <HTML> e </HTML> devem englobar todo o conteúdo da página (estar presente no início e no fim do texto) para indicar ao navegador que se trata de um documento HTML. O documento, por sua vez, está dividido em duas partes: o cabeçalho e o corpo do texto, cada um indicado por um par de tags diferente. Tudo que estiver entre o par <HEAD> e </HEAD> irão compor o cabeçalho, não aparecendo na página. O elemento principal do cabeçalho é o título do documento, que deve ser colocado entre o par <TITLE> e </TITLE>. Os navegadores mostram o título na barra de título do programa e na área em aparecem as páginas já visitadas. Por fim, existe o par <BODY> e </BODY>, que serve para indicar o corpo do texto, ou seja, a parte mostrada na janela do navegador.

Um formulário é um modelo para a entrada de um conjunto de dados, esses dados são inseridos através de janelas de entrada de dados. Para depois trabalharmos com os scripts, que são os programas que manipulam esses dados. Oferecendo os serviços desejados como, acesso a banco de dados, envio de e-mail, entre outros.

O elemento <FORM> delimita um formulário e contém uma sequência de elementos de entrada e de formatação do documento, e o campo <INPUT> tem um atributo TYPE, ao qual podemos atribuir seis valores diferentes para gerar seis tipos diferentes de entrada de dados.

Quando INPUT não apresenta atributos, é assumido TYPE do tipo TEXT (default), como mostram as Figuras 3 e 4.

```
<FORM>
Nome: <INPUT TYPE=TEXT NAME="Nome">
</FORM>
```

Figura 3 – código do formulário tipo TEXT

Nome:

Figura 4 – Resultado

Quando na entrada de texto os caracteres são escondidos por asteriscos, o TYPE é do tipo PASSWORD, como se pode ver nas Figuras 5 e 6.

```
<FORM>
Login: <INPUT TYPE=TEXT NAME="login"><br>
Password: <INPUT TYPE=PASSWORD
NAME="senha">
</FORM>
```

Figura 5 – código do formulário tipo PASSWORD

Login:

Rodrigo

Password:

Figura 6 – Resultado

Quando as opções são de múltipla escolha, o TYPE é do tipo CHECKBOX, como mostram as Figuras 7 e 8. Cada alternativa corresponde a um valor a ser manipulado pelo *script*, que processa os dados.

```
Esportes que você pratica:
<INPUT TYPE=CHECKBOX NAME="esporte"
VALUE="futebol">Futebol<br>
<INPUT TYPE=CHECKBOX NAME="esporte"
VALUE="basquete">Basquete <br >
<INPUT TYPE=CHECKBOX NAME="esporte"
VALUE="bocha">Bocha
```

Figura 7 – código do formulário tipo CHECKBOX

Esportes que você pratica:

☐

Futebol

☒

Basquete

☒

Bocha

Figura 8 – Resultado

Quando a entrada de dados é feita através de opção única, isto é, quando somente uma alternativa pode ser escolhida, o TYPE é do tipo RADIO, como mostram as Figuras 9 e 10.

```
Qual o pior time do Brasil:
<FORM>
<INPUT TYPE=RADIO NAME="time"
VALUE="palm">Palmeiras<br>
<INPUT TYPE=RADIO NAME="time"
VALUE="fla">Flamengo<br>
<INPUT TYPE=RADIO NAME="time"
VALUE="san">Santos <br>
<INPUT TYPE=RADIO NAME="time"
VALUE="inte">Internacional
</FORM>
```

Figura 9 – código do formulário do tipo RADIO

```
Qual o pior time do
Brasil:
☒ Palmeiras;
☐ Flamengo;
☐ Santos;
☐ Internacional;
```

Figura 10 – Resultado

Quando a entrada de dados é feita através de botões de ação, isto é, quando é apresentado o botão que causa o envio dos dados de entrada para o servidor, o TYPE é do tipo SUBMIT, como mostram as Figuras 11 e 12.

```
<FORM>
<INPUT TYPE=SUBMIT VALUE="Envia mensagem">
</FORM>
```

Figura 11 – código do formulário do tipo SUBMIT

Envia mensagem

Figura 12 – Resultado

Quando a entrada de dados é feita através do SELECT, que apresenta uma lista de valores através de campos OPTION, o TYPE é do tipo SELECT, como mostram as Figuras 13 e 14.

```
<SELECT NAME="sabor">
<OPTION>Abacaxi
<OPTION>Creme
<OPTION>Morango
<OPTION>Chocolate
</SELECT>
```

Figura 13 – código do formulário do tipo SELECT

Abacaxi
Abacaxi
Creme
Morango
Chocolate

Figura 14 – Resultado

2.2.2 XML

A XML é uma linguagem que gera um formato para descrever dados estruturados, permitindo a definição de um número infinito de tags, partindo do mesmo princípio da Linguagem de Marcação de Hipertexto (HTML). Basicamente o surgimento da XML se deu para solucionar os problemas de limitações da HTML e porque a HTML não conseguiu acompanhar as novas demandas de mercado.

A XML não se concentra na aparência, mas sim na estrutura. Ela é extensível porque não predefine *tag* nenhum, mas possibilita ao autor criar tags que sejam necessários conforme a sua aplicação.

Foi criada uma especificação preliminar de uma linguagem para apresentação dos documentos XML, conhecida por XLS – *Extensible Style Language* (Linguagem de Estilo Extensível), como também existe uma linguagem para apresentação de documentos HTML denominada CSS – *Casting Style Sheet* (Folhas de Estilo).

A capacidade da XLS não se restringe apenas a fornecer recursos para formatar um documento na tela do *browser*. É possível fazer com que o mesmo documento seja visualizado de formas diferentes, sendo que seus elementos podem ser filtrados, exibidos segundo certas condições, ordenados segundo determinadas regras (Furgeri, 2001).

Uma aplicação XML é exemplificada na Figura 15.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<catálogo>
  <item>
    <nome>José da Silva</nome>
    <endereço>
      <rua>Praça da Aranha, 17</rua>
      <estado>SP</estado>
      <código-postal>30850-000</código-postal>
      <cidade>São Paulo</cidade>
      <país>Brasil</país>
    </endereço>
    <tel preferido="sim">11-234-6558</tel>
    <tel>11-223-7198</tel>
  </item>
  <item>
    <nome><pnome>Carlos</pnome>
    <snome>dos Santos</snome></nome>
    <tel>11-233-5446</tel>
  </item>
</catálogo>
```

Figura 15 – Aplicação XML

2.2.3 JAVA

A linguagem de programação JAVA, desenvolvida pela *Sun Microsystem* em 1991, foi baseada nas linguagens C e C++ e foi projetada para ser uma linguagem de programação independente de máquina, segura, poderosa e portátil para qualquer plataforma bastando apenas que se porte o interpretador.

JAVA obteve sucesso porque:

- Os sistemas em JAVA são facilmente expressados e entendidos;
- Aplicações maximizadas com o uso de bibliotecas de classes;
- Manipulação de erros;
- Própria para programação em larga escala.

Segundo Niemeyer e Knudsen (2000), a JAVA oferece várias camadas de proteção contra um código perigosamente danificado, além de elementos perigosos como vírus e cavalos de tróia, tornando-se assim uma linguagem bastante segura.

Programas em JAVA são basicamente compostos por classes, métodos, onde são processados as operações, e atributos, que são variáveis de armazenamento temporário. A JAVA contém muitas classes predefinidas que são agrupadas por diretórios em disco segundo categorias de classes relacionadas chamadas pacotes.

Um dos componentes que JAVA oferece, bastante utilizados pelos desenvolvedores, é o *JavaBean* (muitas vezes chamados simplesmente de *beans*). Este componente de aplicação, esta originalmente ligado ao desenvolvimento de aplicações visuais, como botões, caixas de texto e outros elementos necessários ao desenvolvimento de uma interface visual para um programa qualquer.

Os ambientes de programação e projeto gráficos que suportam *beans*, fornecem aos programadores muita flexibilidade ao permitir que eles reutilizem e integrem diferentes componentes existentes, que em muitos casos não foram concebidos para serem utilizados em conjunto.

Em qualquer aplicativo é apropriado que a maioria ou todas as informações sejam armazenadas em locais seguros, para que os dados sejam atualizados e localizados rapidamente. O sistema de banco de dados não apenas

forneçam esses recursos como também organiza as informações de maneira que facilita a realização de consultas.

Quando uma aplicação JAVA precisa se comunicar com banco de dados, ela o faz através de uma API de Conectividade de Banco de Dados JAVA (JAVA Database *Connectivity* – JDBC) de classes (Java.sql) que permite se trabalhar com banco de dados.

2.2.4 JSP

A *Java Server Pages* (JSP) é uma linguagem baseada em JAVA que simplifica o processo de desenvolvimento de *sites* da *web* dinâmicos e interativos.

Segundo Sieldf e Kolb (2000), JSP oferece diversos benefícios, um deles é por ela ser baseada na linguagem JAVA, portanto, JSP herda todos os pontos positivos desta tecnologia poderosa, como por exemplo: encapsulamento, exceções, gerenciamento de memória, entre outras.

Devido à sua habilidade em separar a lógica de implementação da apresentação e a portabilidade de plataforma, podendo ser executado em outros sistemas operacionais além dos da *Microsoft*. O JSP vem para substituir as páginas de formato tradicionais como ASP. A linguagem JSP conseguiu isto ao combinar texto de marcação padrão com elementos de criação de *scripts* e componentes orientados a objetos.

Servlets são componentes, pequenos programas na linguagem Java, que geram as páginas visualizadas pelo navegador. Eles podem conter a lógica e a apresentação do conteúdo de uma apresentação ou podem funcionar apenas como componentes da aplicação, dando suporte a outros *servlets* ou a páginas JSP.

Quando o cliente faz a solicitação de um arquivo JSP, é enviado um pacote de requisição para o servidor JSP, que invoca o *servlet* compilador de página para tratar da solicitação. O servidor JSP envia a solicitação de qualquer componente especificado no arquivo. O componente controla a requisição possibilitando a recuperação de arquivos em banco de dados relacional, em seguida, passa o objeto resposta de volta para servidor JSP. O servidor envia a página JSP revisada de volta para o cliente, onde o usuário pode visualizar os resultados através do *browser*, como ilustra a Figura 16.

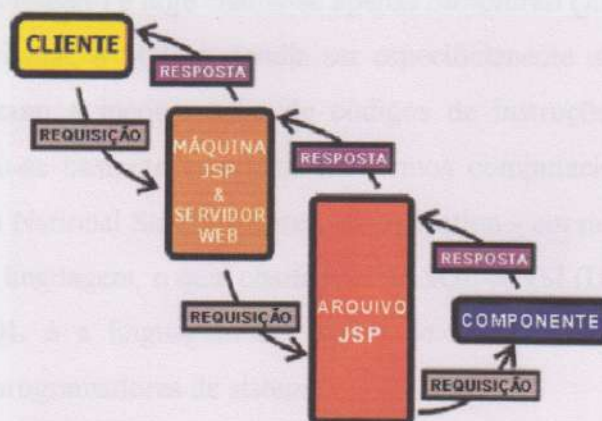


Figura 16 – Descrição do processamento Cliente/Servidor.

A seguir será representado uma aplicação JSP, onde mescla *tags* predefinidas do HTML, com *tags* personalizadas JSP e com acesso a um banco de dados relacional.

```

<html><title>Database Exemplo</title>
<head>
  <%@ taglib uri='/WEB-INF/tlds/database.tld' prefix='database'%>
</head>
<body>
<database:connect database='F:/database/sunpress'>
  <database:query>
    SELECT * FROM Customers, Orders
  </database:query>
  <table border='2' cellpadding='5'>
    <database:columnNames columnName='column_name'>
      <th><%=column_name %> </th>
    </database:columnNames>
    <database:rows><tr>
      <tr><database:columns columnsValue='column_value'>
        <td><%=column_value %></td>
      </database:columns></tr>
    </database:rows>
  </table>
</database:connect>
</body>
</html>
  
```

Figura 17 – Aplicação JSP

2.2.5 SQL

A SQL foi desenvolvida originalmente no laboratório de pesquisa da IBM *Research* no início da década de 70. Chamava-se inicialmente SEQUEL (*Structured*

English Query Language) e hoje chama-se apenas *Structured Query Language* (SQL). Em sua forma original, a SQL pretendia ser especificamente uma “sublinguagem de dados”, porém, com a incorporação de códigos de instruções e declarações esta ferramenta tornou-se bastante completa em termos computacionais. Visando isto a ANSI - American National Standard Internationalization - em meados de 86, publicou um padrão para a linguagem, o qual chamamos de SQL-ANSI (Date, 2000).

A SQL é a linguagem de Banco de Dados orientada à objetos mais popular entre os programadores de sistemas.

Para Soukup (1998), a SQL foi projetado para suportar processamento de transações de grande volume, bem como para armazenamento de dados e aplicativos de apoio de decisão.

A SQL possui algumas operações básicas que estão citadas a seguir:

- **Extração de informações do banco de dados**

Um banco de dados relacional consiste de uma coleção de relações, cada uma designada por um único nome. Uma consulta ao banco de dados SQL, que seleciona um ou mais campos de uma ou mais tabelas, pode ser efetuada através da estrutura **SELECT**.

- **Select:** Usada para relacionar os atributos necessários em uma consulta;
- **From:** Associa as relações que serão pesquisadas durante a evolução de uma expressão;
- **Where:** Consiste em um predicado envolvendo atributos da relação que aparece na da cláusula From.

Exemplo:

```
SELECT Endereco, Telefone FROM Cliente WHERE Nome = “Rodrigo”;
```

- **Modificações no banco de dados**

Expressamos as modificações (inclusão, alteração e exclusão) no banco de dados SQL usando basicamente as estruturas **INSERT**, **UPDATE** e **DELETE**, que serão detalhadas em seguida (Silberschatz, 1999)

- Incluir

Para inserir dados em uma relação, pode-se especificar um registro a ser inserido ou escrever uma consulta cujo resultado seja um conjunto de registros a inserir. Certamente, os valores dos atributos para os registros a inserir devem pertencer ao domínio desses atributos, bem como, a ordem correta dos registros.

- Exemplo:**
- Insert into: Comando usado para inserir dados em um registro;
 - Values: especificação de qual dados serão inseridos.

Exemplo:

```
INSERT INTO conta(nome_ag, num_conta, saldo)
VALUES ("Banco do Brasil Centro", "A-9732", 1200)
```

- Excluir

Pode-se remover somente registros inteiros, não podendo entretanto excluir valores de um atributo em particular. Caso precise-se remover registros de diversas relações, o comando de remoção para cada relação deverá ser usado.

- Delete: Comando usado para excluir dados em um registro;
- From: Associa as relações que serão pesquisadas durante a evolução de uma expressão;
- Where: Consiste em um predicado envolvendo atributos da relação que aparece na cláusula From.

Exemplo:

```
DELETE FROM depositante
WHERE nome_cliente = "Rodrigo Pires"
```

- Alterar

Em algumas situações pode-se querer modificar os valores de determinados campos de um registro, preservando os demais. Com esse propósito usa-se o UPDATE. Assim como o INSERT e o DELETE, pode-se escolher os campos de cada registro que serão atualizadas.

- Update: especificação de qual tabela será feito a atualização;
- Set: a operação, que irá alterar os registros;
- Where: Condição para que o comando seja executado com sucesso.

Exemplo:

UPDATE conta

SET saldo = saldo * 1,05

WHERE saldo <= 10000

3. Implementação de um Protótipo voltado para Comércio Eletrônico

A análise na qual este sistema irá basear-se está representada, de forma clara, pela notação utilizada na linguagem UML (Linguagem de Modelagem Unificada), voltada para a modelagem orientada a objetos.

Utiliza-se, para tal representação, diagramas de caso de uso, descrição dos cenários, diagramas de colaboração e diagrama de classe, entre outros, com a finalidade de ilustrar as funcionalidades do sistema.

A estratégia utilizada para a implementação do sistema consistiu em uma pesquisa dos principais *sites* de comércio eletrônico, para que pudessem ser identificadas todas as informações operacionais e tendências tecnológicas que, posteriormente, viriam a ser utilizadas.

Dentre essas tecnologias, foram escolhidas:

- Linguagem de marcação de texto: HTML 4.01;
- Linguagem estruturada: XML 1.0;
- Linguagens de aplicação: JAVA 2, JSP 1.2;
- Banco de Dados: SQL 6.5.

O Ambiente de desenvolvimento utilizado para a implementação desse sistema foi o Forte, que utiliza vários componentes integrados.

Um desses componentes é o J2EE, ambiente Java v.2 Enterprise Edition, que possui algumas ferramentas para facilitar o desenvolvimento, como o web server Tomcat 3.2 para compilação e manipulação dos demais componentes necessários.

Outros componentes são: o SAX, documento manipulador de acesso ao XML que permite a interação com Java e o DTD, um tipo de documento que descreve e organiza as Tags que podem ser usadas no arquivo XML. O arquivo DTD pode ser gerado a partir de um arquivo XML, ou ainda pode ser gerado um arquivo DTD vazio definido pelo código fonte da página em questão.

Essas escolhas proporcionaram ao sistema algumas características relevantes, dentre elas:

- Portabilidade: o sistema pode ser facilmente distribuído, por ter sido planejado para ser executado na Internet, independente da plataforma operacional;

- Coerência dos dados: esta relacionada a garantia do envio e recebimento dos dados íntegros. A centralização do banco de dados tenta garantir essa coerência;
- Acessibilidade: os usuários do sistema, para acessá-lo, necessitam apenas de um terminal com acesso a Internet e um *browser*;
- Baixo custo: o fator custo é determinante para o sucesso, aceitação e distribuição de sistemas computacionais. Foram utilizados módulos computacionais simples e de larga aceitação para o desenvolvimento do ambiente proposto.

3.1 Diagrama de Classe

O diagrama de classes do projeto ilustra as especificações para as classes de *software* e de interfaces. As informações, mostradas na Figura 17, incluem as classes, associações, atributos, métodos, dependência e cardinalidade.

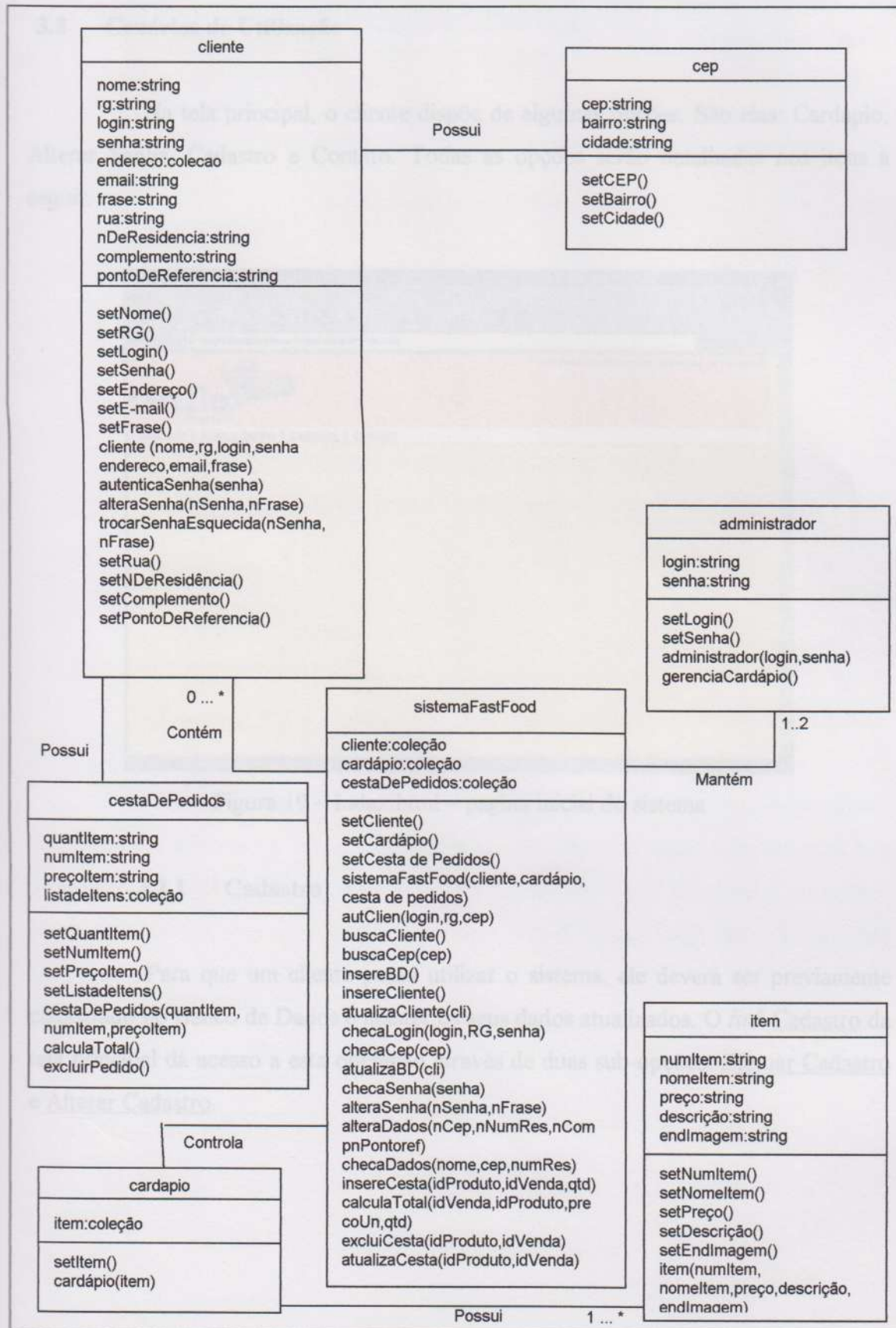


Figura 18 – Diagrama de Classes

3.2 Cenários de Utilização

Na tela principal, o cliente dispõe de algumas opções. São elas: Cardápio, Alterar Senha, Cadastro e Contato. Todas as opções serão detalhadas nos itens a seguir.

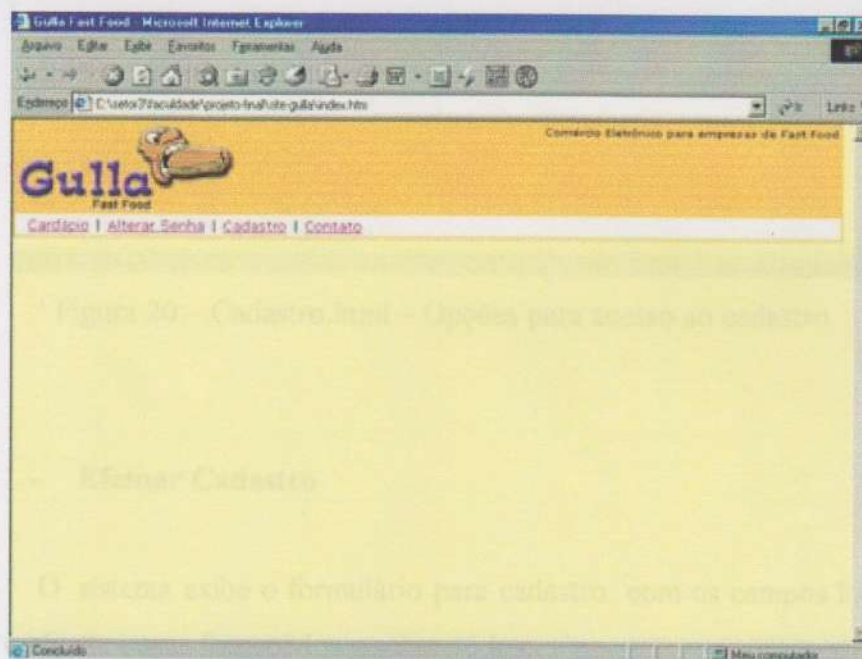


Figura 19 – Index.html – pagina inicial do sistema

3.2.1 Cadastro

Para que um cliente possa utilizar o sistema, ele deverá ser previamente cadastrado no Banco de Dados e manter os seus dados atualizados. O link Cadastro da tela principal dá acesso a esta operação através de duas sub-opções: Efetuar Cadastro e Alterar Cadastro.

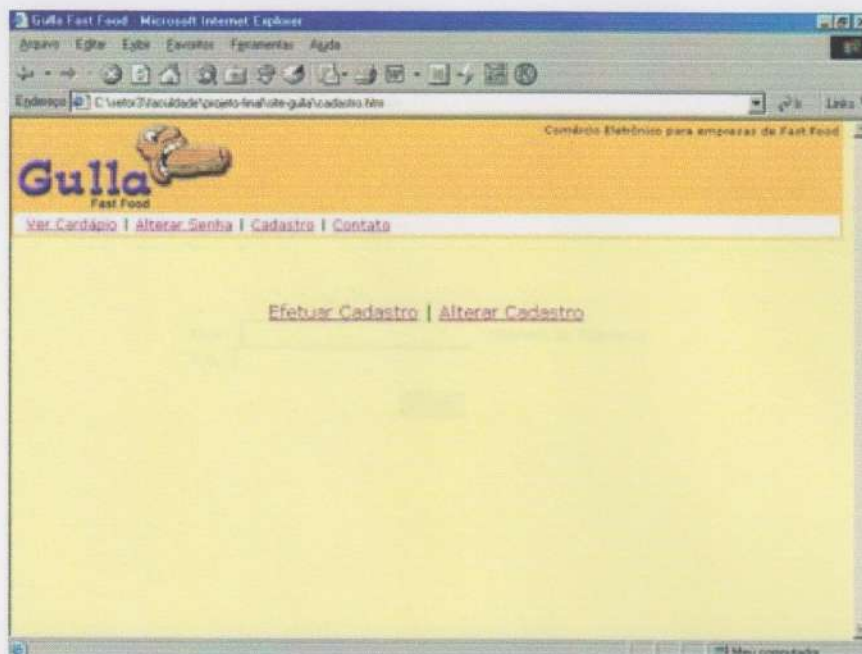


Figura 20 – Cadastro.html – Opções para acesso ao cadastro

- Efetuar Cadastro

O sistema exibe o formulário para cadastro, com os campos login e R.G, para que o cliente possa fornecê-los e submetê-los.

Em seguida, o sistema fará uma consulta para verificar se já existe algum cliente cadastrado com o mesmo login e R.G.. Se existir, o sistema exibirá uma tela informando que já existe um cadastro com aquele login e R.G.. Senão, o sistema valida o login e R.G e exibe outro formulário com o campo CEP, que é preenchido pelo cliente.

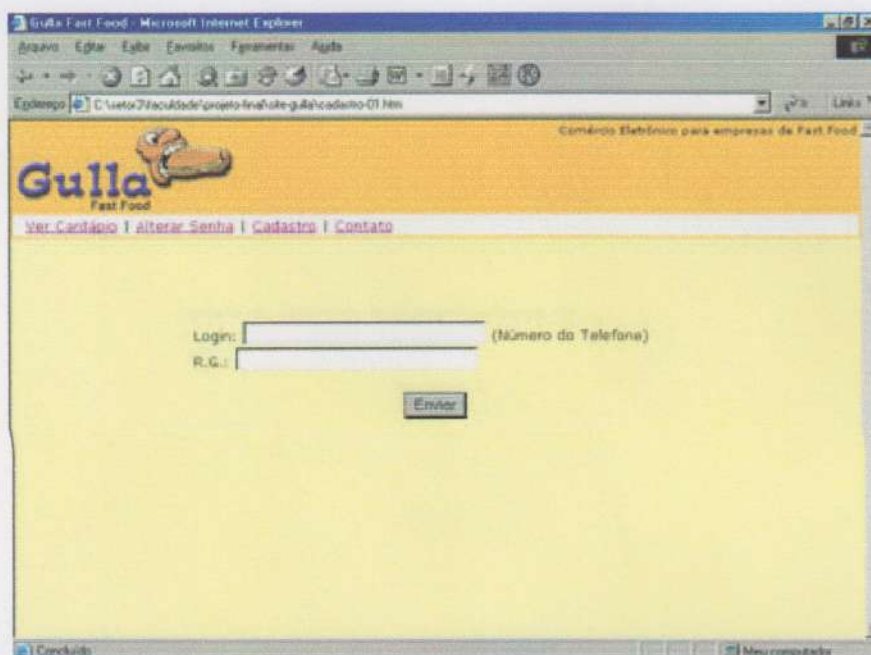


Figura 21 – Identific.html – Requisição Identificação

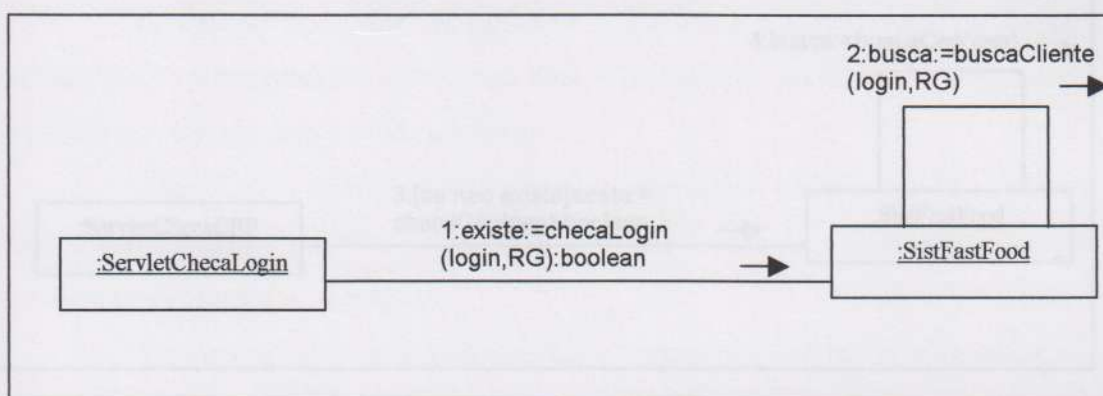


Figura 22- Diagrama de Colaboração ChecaLogin

O sistema checa o CEP e, se for válido, exibe a mensagem de que a localidade é válida, disponibilizando em seguida um formulário com os campos: nome completo, número da residência, complemento, ponto de referência, e-mail. Se o CEP não for válido, o sistema exibirá uma tela informando que o CEP digitado não confere ou não faz parte da área de cobertura de entregas da empresa.

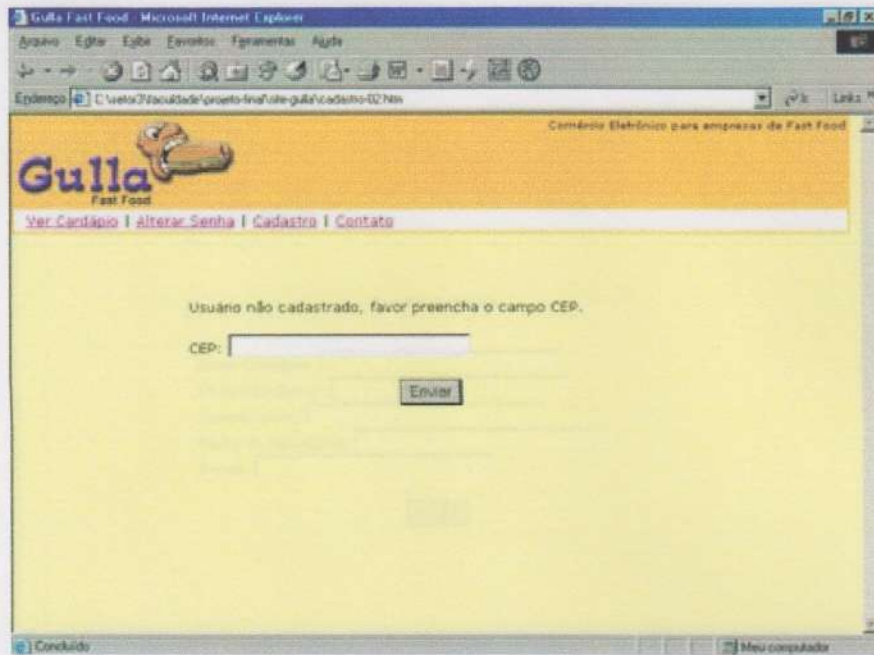


Figura 23 – `checaCep.jsp` – Requisição CEP

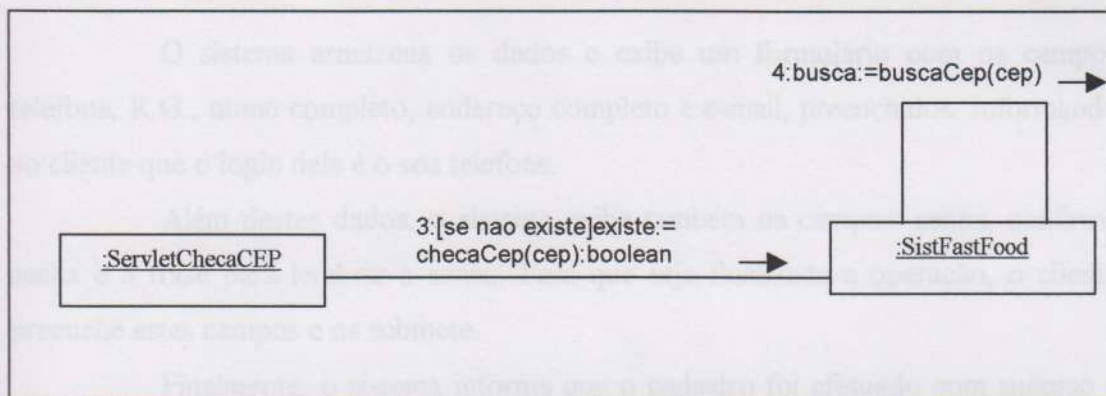


Figura 24 – Diagrama de Colaboração ChecaCEP

O cliente procede com o preenchimento destes dados e os submete.

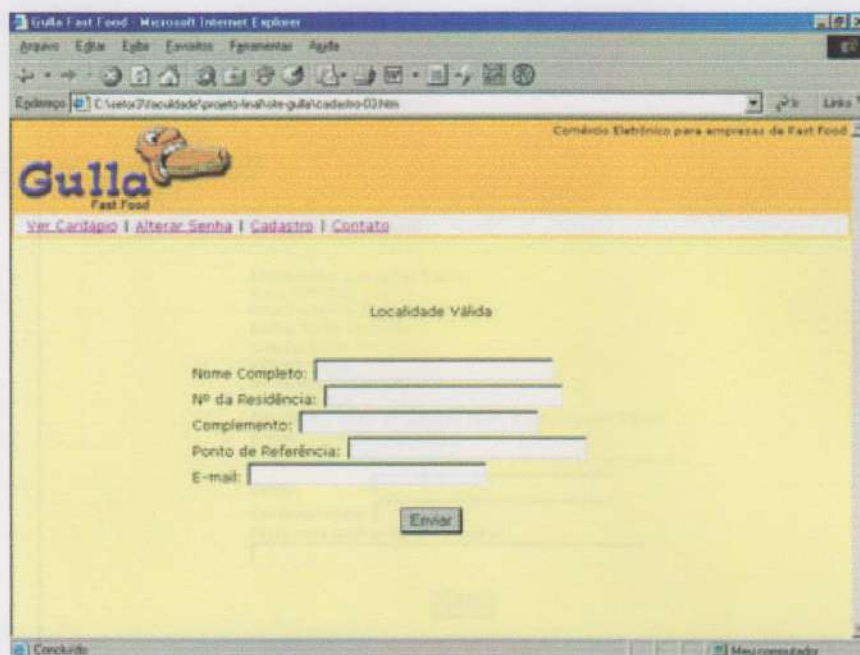


Figura 25 – cadastra.jsp – Requisição Complemento

O sistema armazena os dados e exibe um formulário com os campos telefone, R.G., nome completo, endereço completo e e-mail, preenchidos. Informando ao cliente que o login dele é o seu telefone.

Além destes dados, o sistema exibe também os campos: senha, confirma senha e a frase para lembrar a senha. Para que seja finalizada a operação, o cliente preenche estes campos e os submete.

Finalmente, o sistema informa que o cadastro foi efetuado com sucesso e exibe o *link* Ver Cardápio.

Figura 27 – Página de Cadastro

Gulla Fast Food - Microsoft Internet Explorer

Arquivo Editar Exibir Favoritos Ferramentas Ajuda

Endereço: C:\inetpub\wwwroot\gulla\cadastra-04.htm

Comércio Eletrônico para empresas de Fast Food

Gulla
Fast Food

[Ver Cardápio](#) | [Alterar Senha](#) | [Cadastro](#) | [Contato](#)

Nome: Andre Luis Lopes Santos
R.G.: 1903793
Rua: Professor Clementino Câmara
Bairro: Santo Vermelho
Cidade: Natal
CEP: 59439409
Nº da Residência: 243
Complemento:
Referência: Perto da Casa de Apoio à Criança com Câncer
E-mail: andre@digizap.com.br

Senha:
Confirma Senha:
Frase para lembrar da Nova Senha:

Concluído

Figura 26 – cadastra.jsp – Requisição Senha

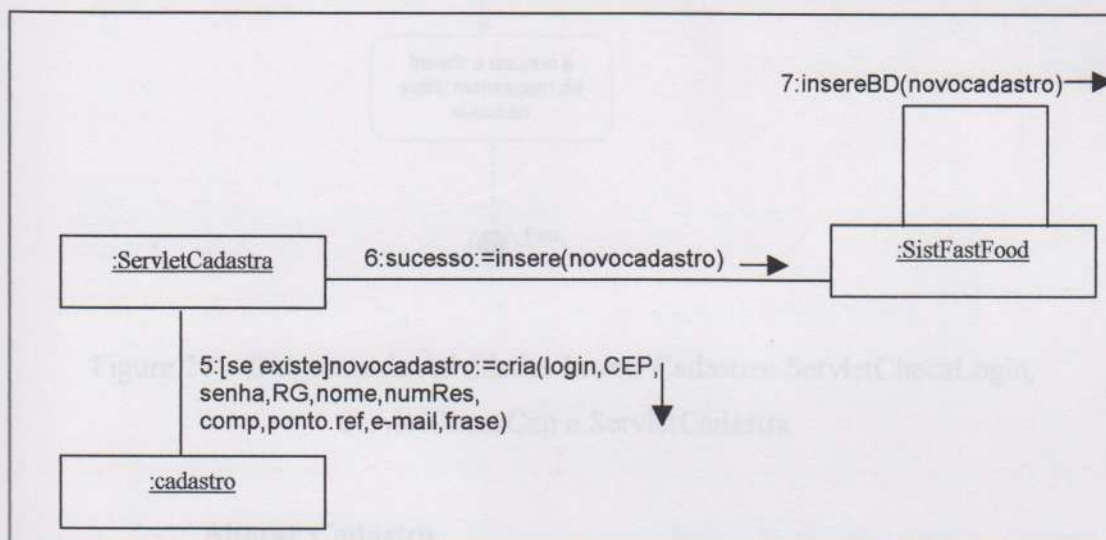


Figura 27 – Diagrama de Colaboração Cadastra

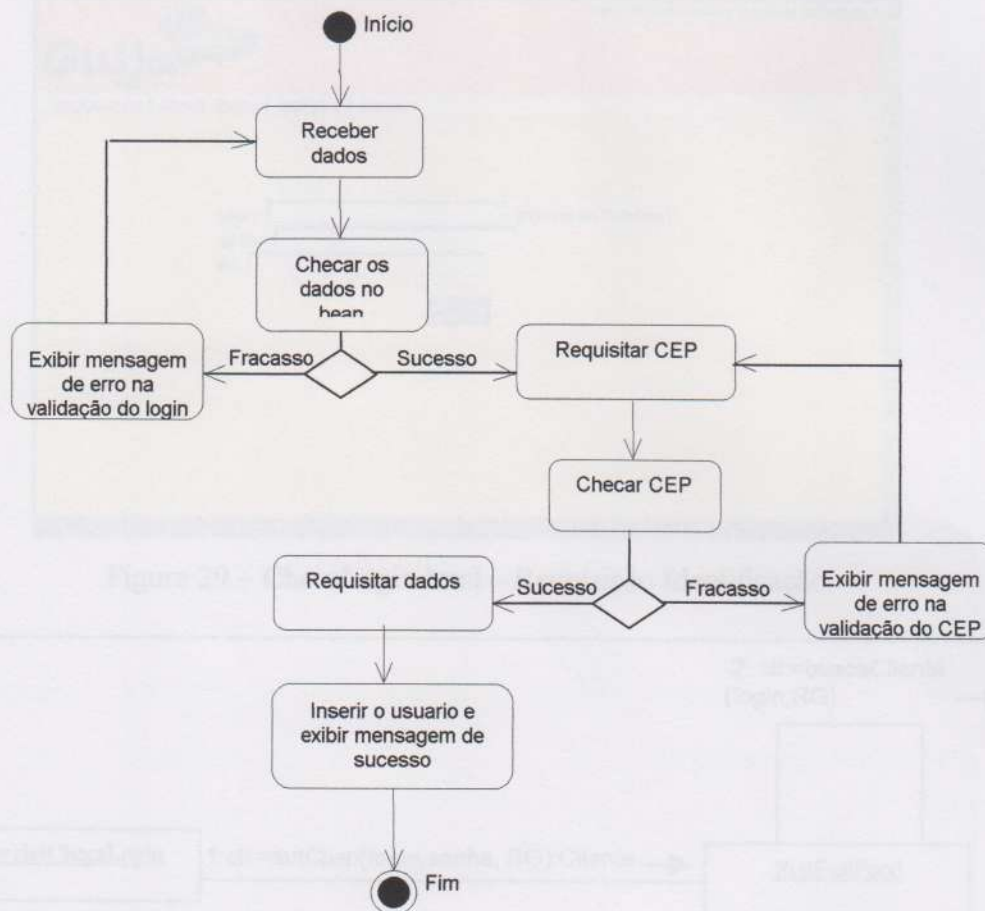


Figura 28 – Diagrama de Atividades Inserir Cadastro: ServletChecaLogin, ServletChecaCep e ServletCadastra

- Alterar Cadastro

Para proceder com a alteração de seu cadastro, o sistema exibe para o cliente um formulário requisitando o seu login (telefone), R.G. e senha. Em seguida, o cliente preenche os campos e submete as informações.

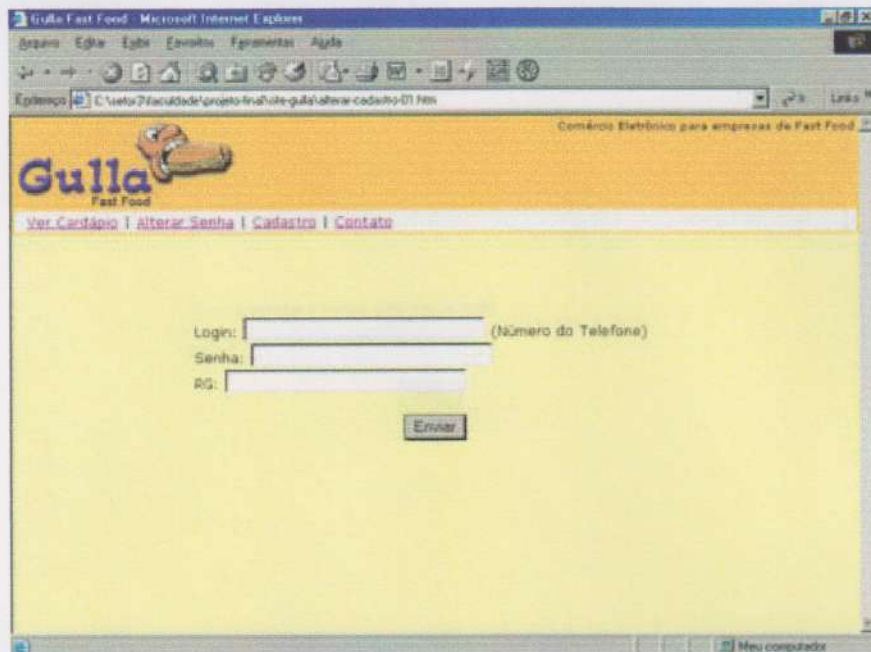


Figura 29 – ChecaLogin.html – Requisição Identificação

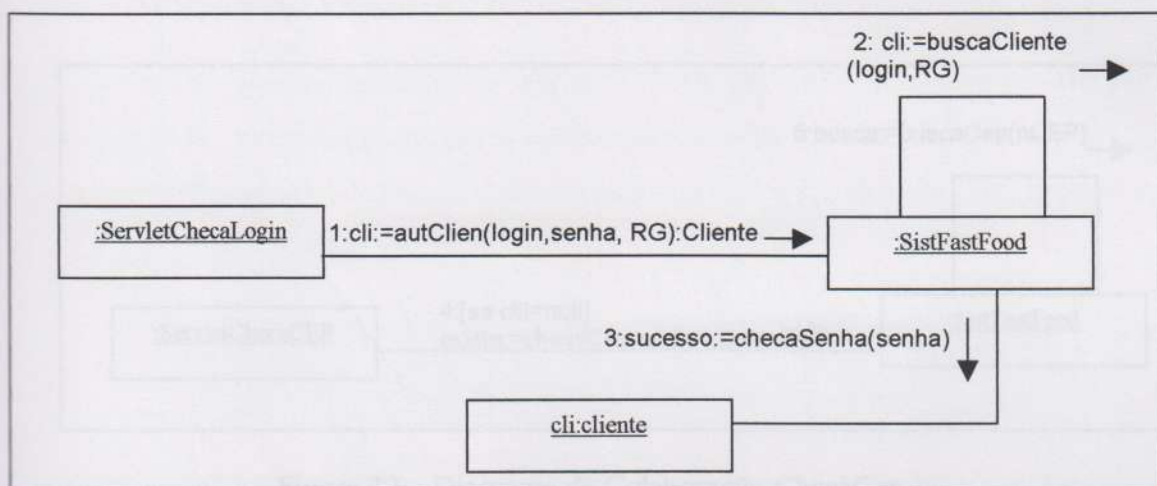


Figura 30 – Diagrama de Colaboração ChecaLogin

O sistema procede então com uma busca no banco de dados.

Se o login não for válido, o sistema exibe uma mensagem de login inválido. Senão, valida o login e requisita o novo CEP, que é informado pelo cliente.

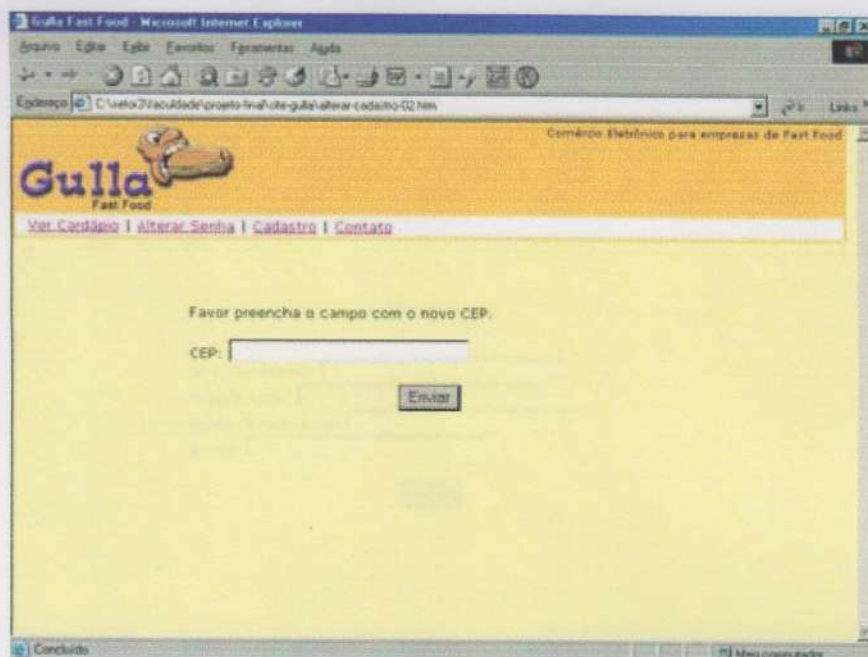


Figura 31 – ChecaCep.jsp – Requisição CEP

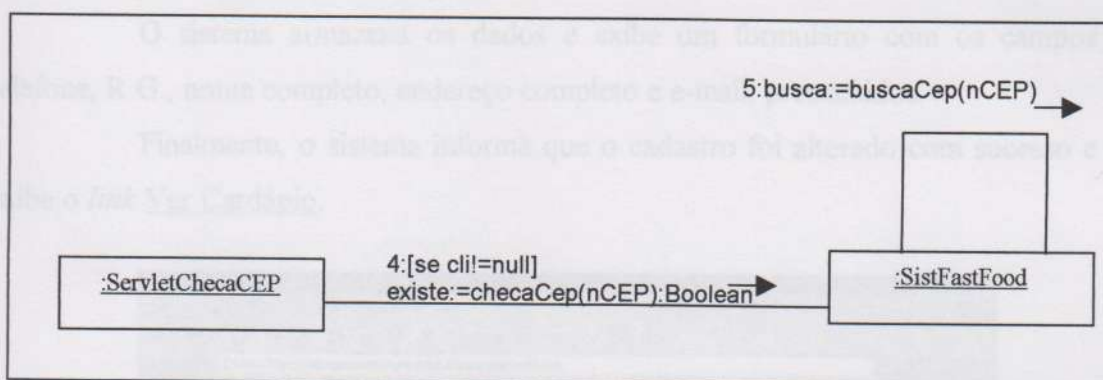


Figura 32 – Diagrama de Colaboração ChecaCep

O sistema checa o CEP e, se for válido, exibe a mensagem de que a localidade é válida, disponibilizando em seguida um formulário com os campos: número da residência, complemento, ponto de referencia, e-mail. Se o CEP não for válido, o sistema exibirá uma tela informando que o CEP digitado não confere ou não faz parte da área de cobertura de entregas da empresa.

O cliente procede com o preenchimento destes dados e os submete.

Gulla Fast Food

Ver Cardápio | Alterar Senha | Cadastro | Contato

Localidade Válida

Nº da Residência:

Complemento:

Ponto de Referência:

E-mail:

Enviar

Figura 33 – alteraCadastro.jsp – Requisição Novo Endereço

O sistema armazena os dados e exibe um formulário com os campos telefone, R.G., nome completo, endereço completo e e-mail, preenchidos.

Finalmente, o sistema informa que o cadastro foi alterado com sucesso e exibe o [link Ver Cardápio](#).

Gulla Fast Food

Ver Cardápio | Alterar Senha | Cadastro | Contato

Nome: andre Luis Lopes Santos
R.G.: 1823733
Rua: Professor Clementino Câmara
Bairro: Barra Vermelho
Cidade: Natal
CEP: 59429409
Nº da Residência: 243
Complemento: Perto da Casa de Apoio à Criança com Câncer
Referência: Perto da Casa de Apoio à Criança com Câncer
E-mail: andre@gigap.com.br

Cadastro atualizado com sucesso.

[Ver Cardápio](#)

Figura 34 – alteraCadastro.jsp – Informa Novo Endereço

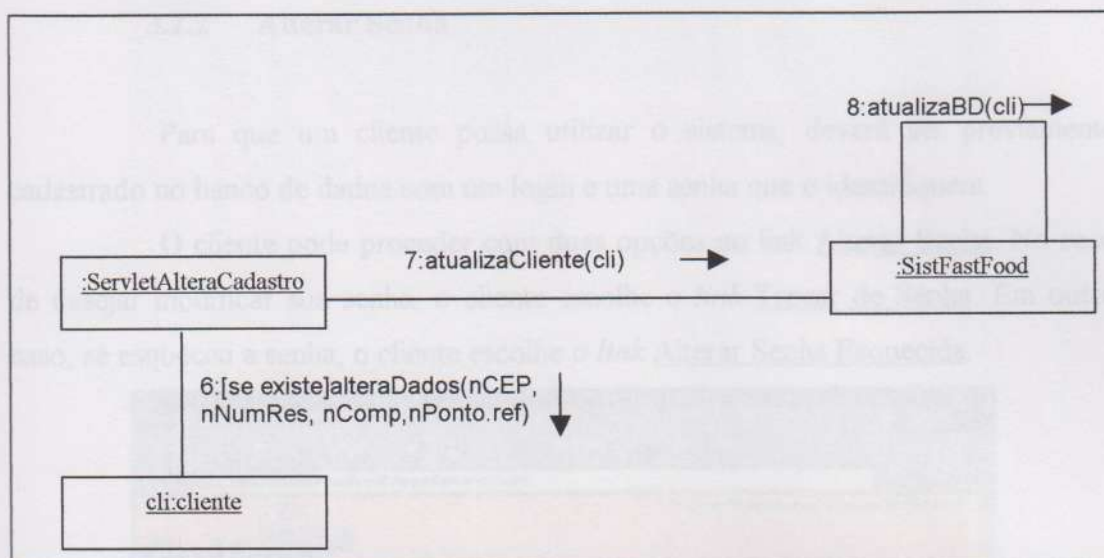


Figura 35 – Diagrama de Colaboração AlteraCadastro

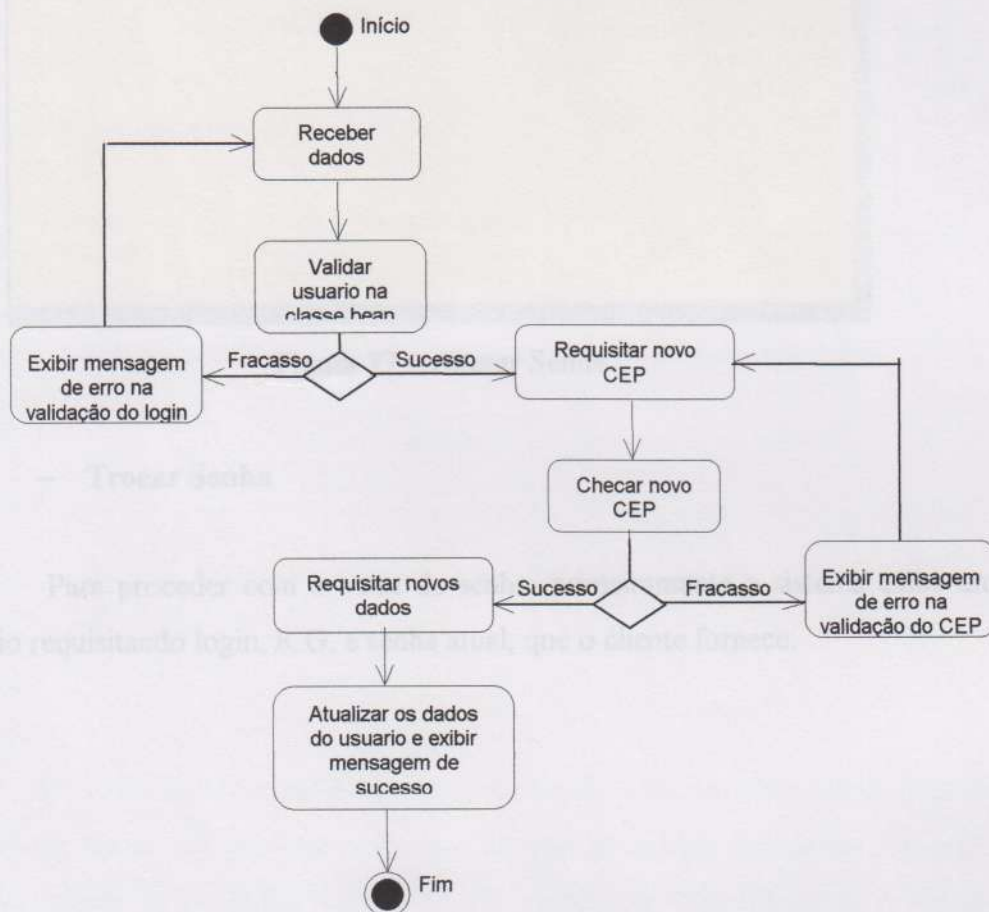


Figura 36 – Diagrama de Atividades Alterar Cadastro: ServletChecaLogin, ServletChecaCep e ServletCadastra

3.2.2 Alterar Senha

Para que um cliente possa utilizar o sistema, deverá ser previamente cadastrado no banco de dados com um login e uma senha que o identifiquem.

O cliente pode proceder com duas opções no link [Alterar Senha](#). No caso de desejar modificar sua senha, o cliente escolhe o *link* [Trocar de Senha](#). Em outro caso, se esqueceu a senha, o cliente escolhe o *link* [Alterar Senha Esquecida](#).

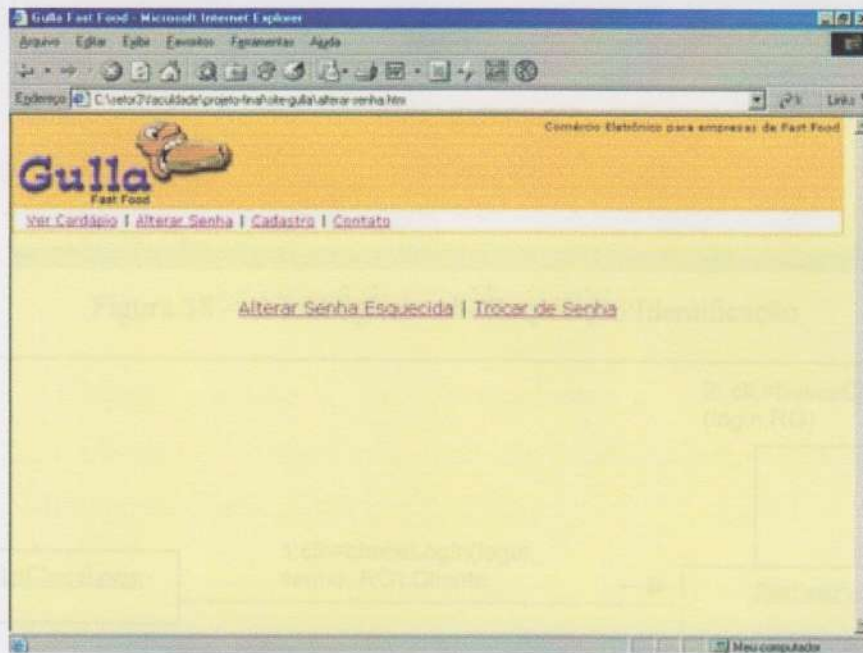


Figura 37 – Alterar Senha

– Trocar Senha

Para proceder com a troca de senha, primeiramente o sistema exibe um formulário requisitando login, R.G. e senha atual, que o cliente fornece.

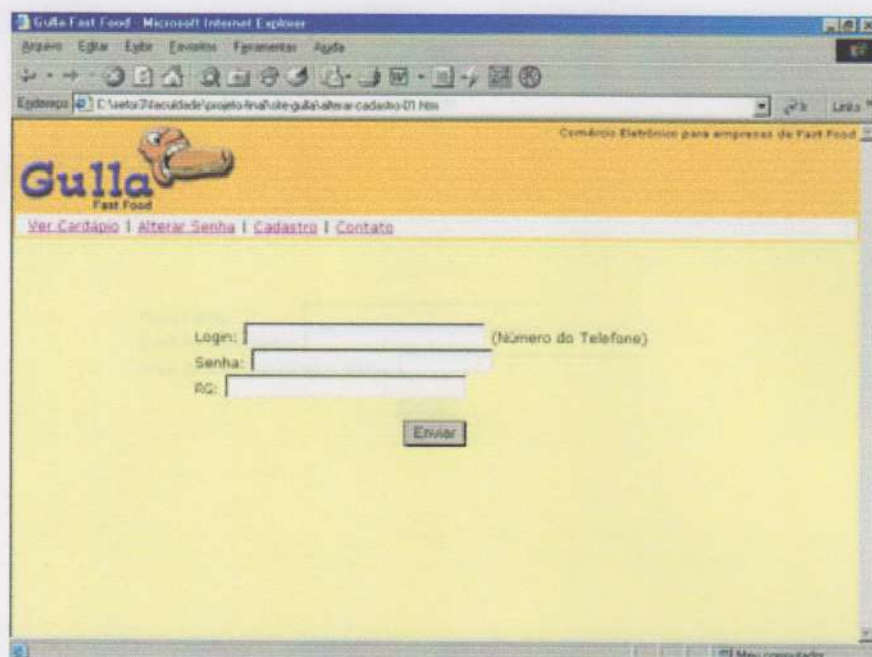


Figura 38 – checkLogin.html - Requisição Identificação

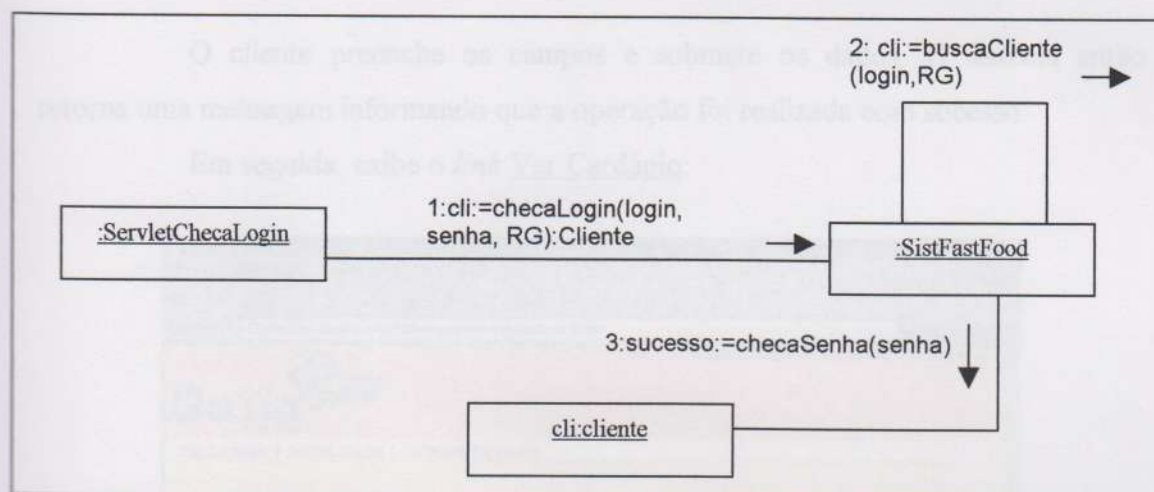


Figura 39 – Diagrama de Colaboracao checkSenha

O sistema então valida estes dados e retorna um formulário com os campos: nova senha, confirma nova senha e formule uma frase para lembrar a senha criada. Entretanto, se os dados login, R.G. e/ou senha não forem válidos, o sistema exibe uma mensagem informando ao cliente o ocorrido.

Figura 40 – alteraSenha.jsp - Requisição Nova Senha

O cliente preenche os campos e submete os dados. O sistema então retorna uma mensagem informando que a operação foi realizada com sucesso.

Em seguida, exibe o *link* [Ver Cardápio](#).

Figura 41 – alteraSenha.jsp – Sucesso Alteração Senha

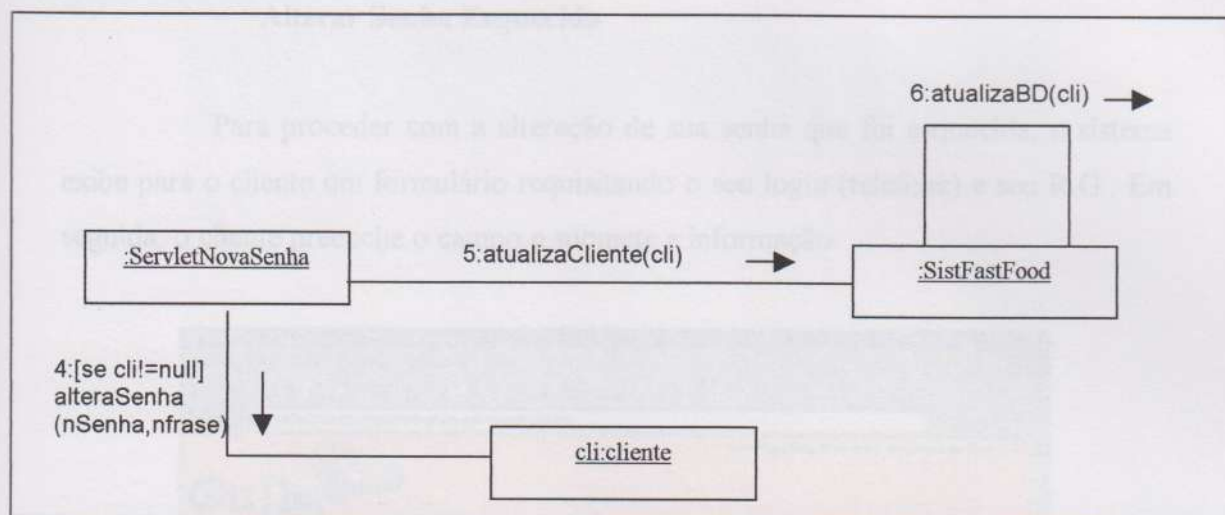


Figura 42 – Diagrama de Colaboração AlteraSenha

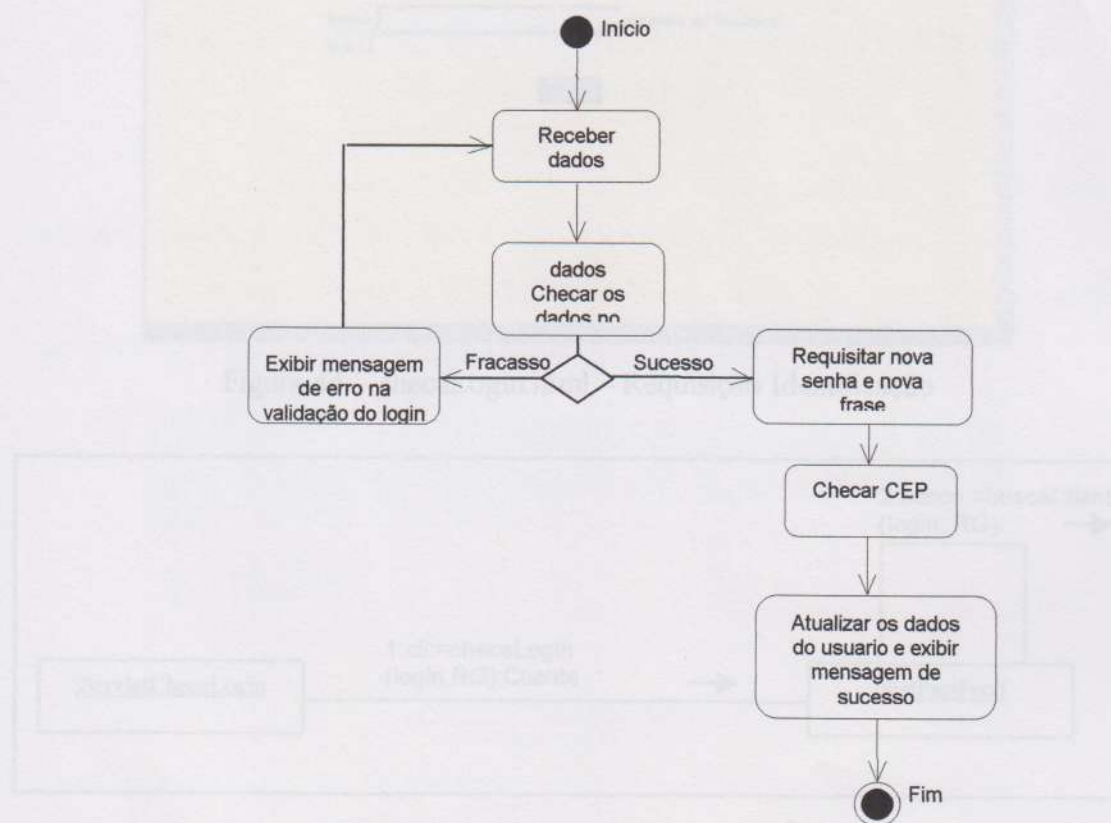


Figura 43 – Diagrama de Atividades Alterar Senha: ServletChecaLogin e ServletNovaSenha

– Alterar Senha Esquecida

Para proceder com a alteração de sua senha que foi esquecida, o sistema exibe para o cliente um formulário requisitando o seu login (telefone) e seu R.G.. Em seguida, o cliente preenche o campo e submete a informação.

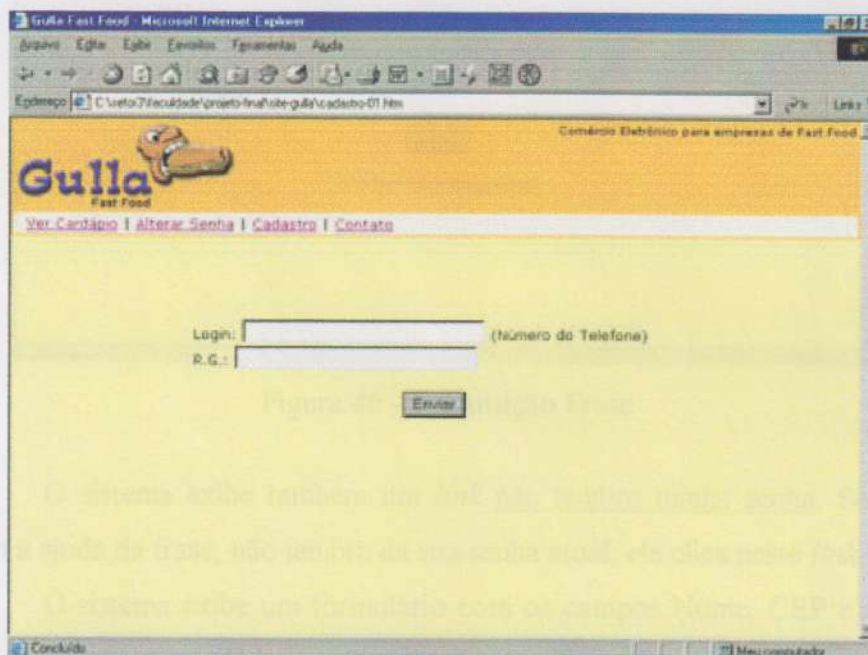


Figura 44 – `checaLogin.html` – Requisição Identificação

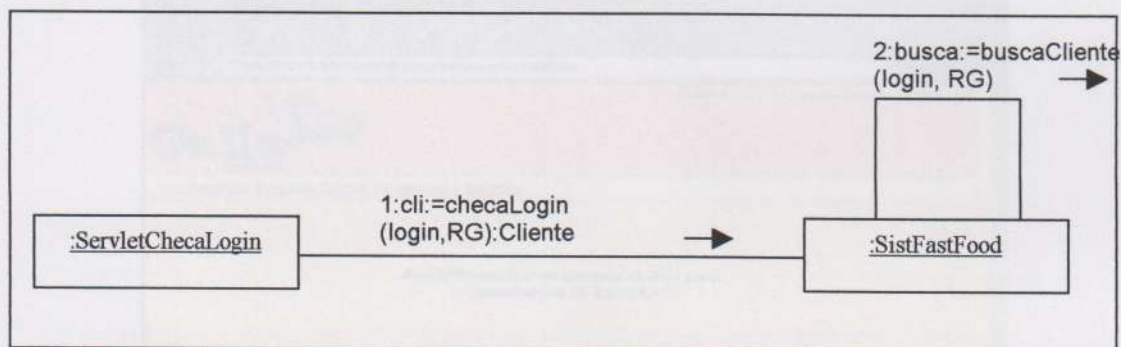


Figura 45 – Diagrama de Colaboração ChecaLogin

O sistema procede então com uma busca no banco de dados.

Se o login e/ou R.G. não for válido, o sistema exibe uma mensagem de operação inválida. Senão, valida a operação e recupera a frase que o cliente digitou para lembrar a senha. Caso o cliente lembre a senha atual, ele a informa, o sistema a autentica e exibe o *link* Ver Cardápio.

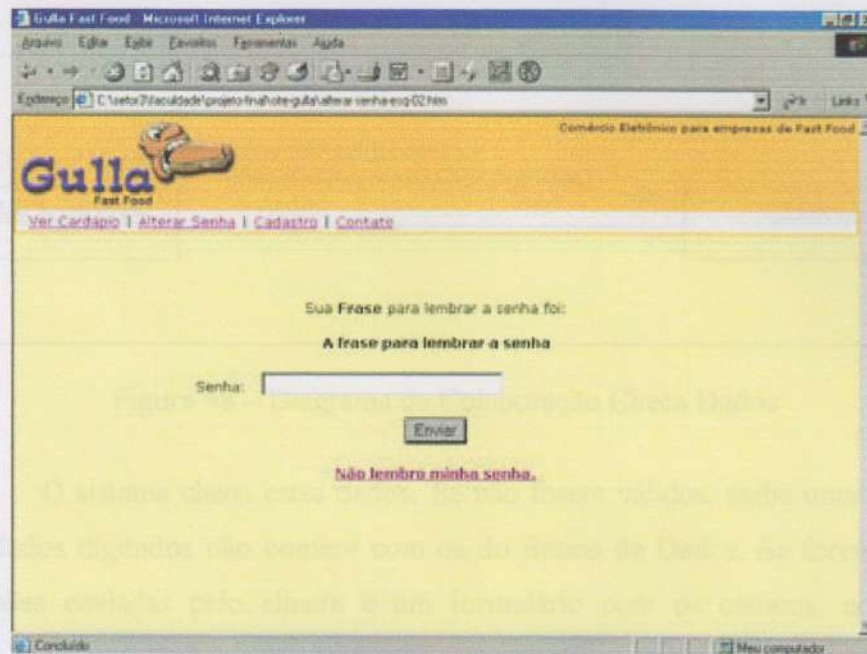


Figura 46 – Requisição Frase

O sistema exibe também um *link* não lembro minha senha. Se o cliente, nem com a ajuda da frase, não lembra da sua senha atual, ele clica neste *link*.

O sistema exibe um formulário com os campos Nome, CEP e número da residência, que são preenchidos e submetidos pelo cliente.

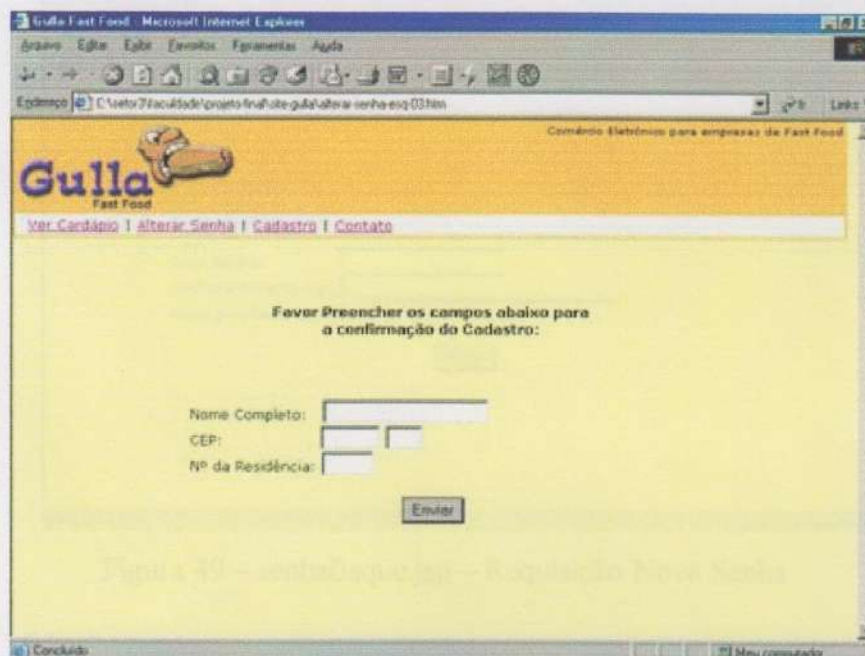


Figura 47 – checkDados.jsp – Confirmação de Dados

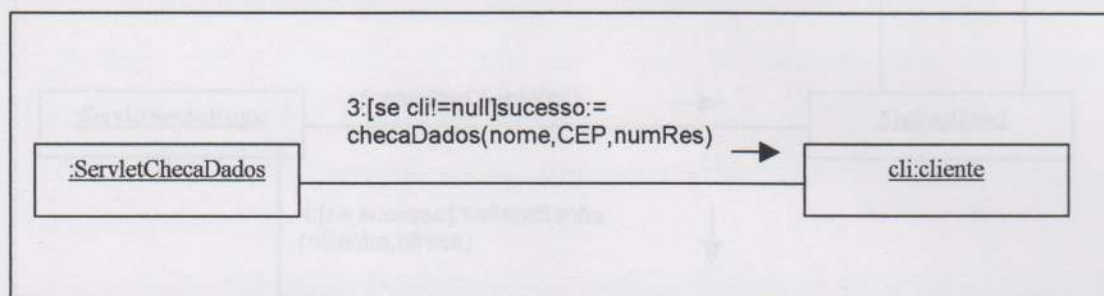


Figura 48 – Diagrama de Colaboração Checa Dados

O sistema checa estes dados. Se não forem válidos, exibe uma mensagem que os dados digitados não confere com os do Banco de Dados. Se forem, exibe as informações enviadas pelo cliente e um formulário com os campos: nova senha, confirma nova senha e frase para lembrar a nova senha.

O cliente preenche o formulário e submete os dados. Finalmente, o sistema armazena os dados, exibe uma mensagem informando que a operação foi realizada com sucesso e disponibiliza o *link* Ver Cardápio.

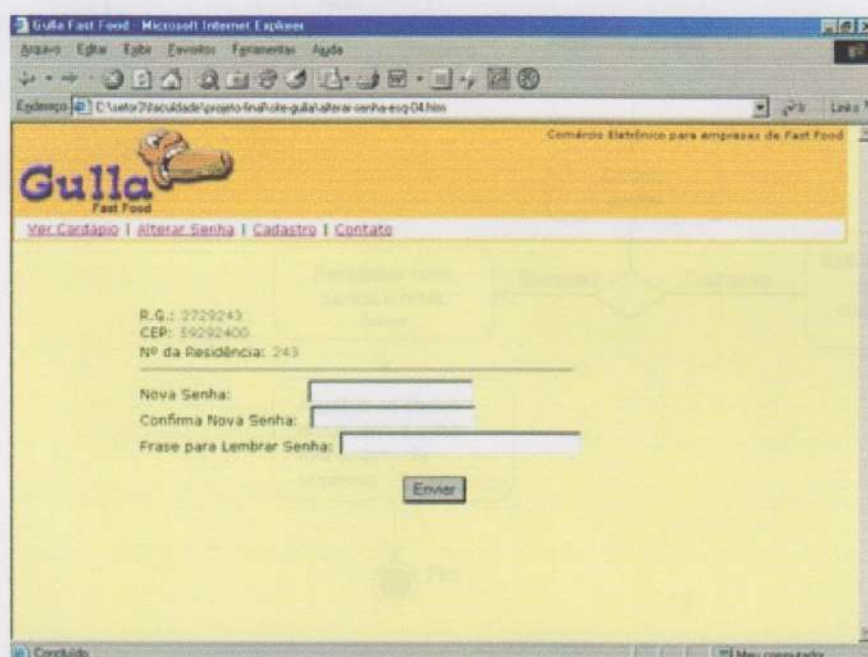


Figura 49 – senhaEsque.jsp – Requisição Nova Senha

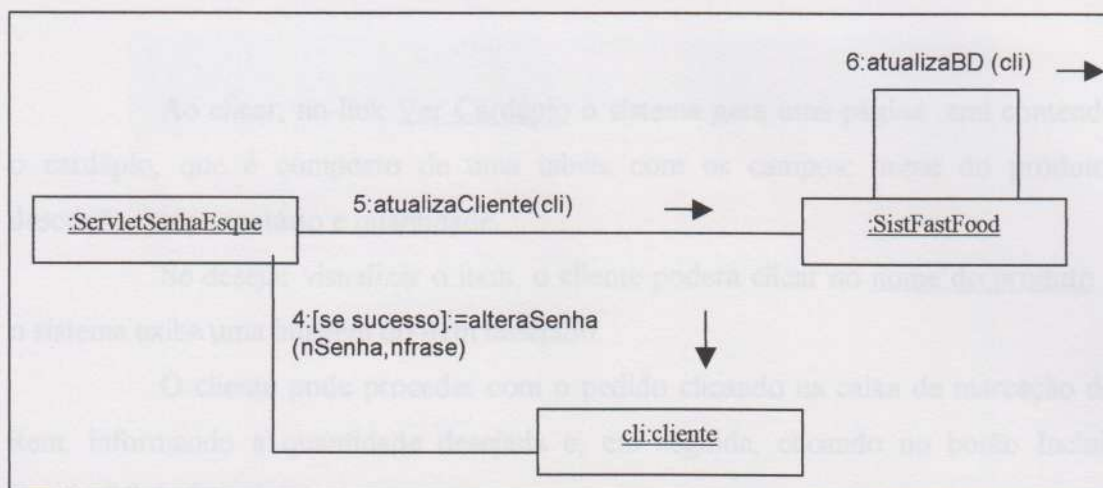


Figura 50 – Diagrama de Colaboração – Requisição Nova Senha

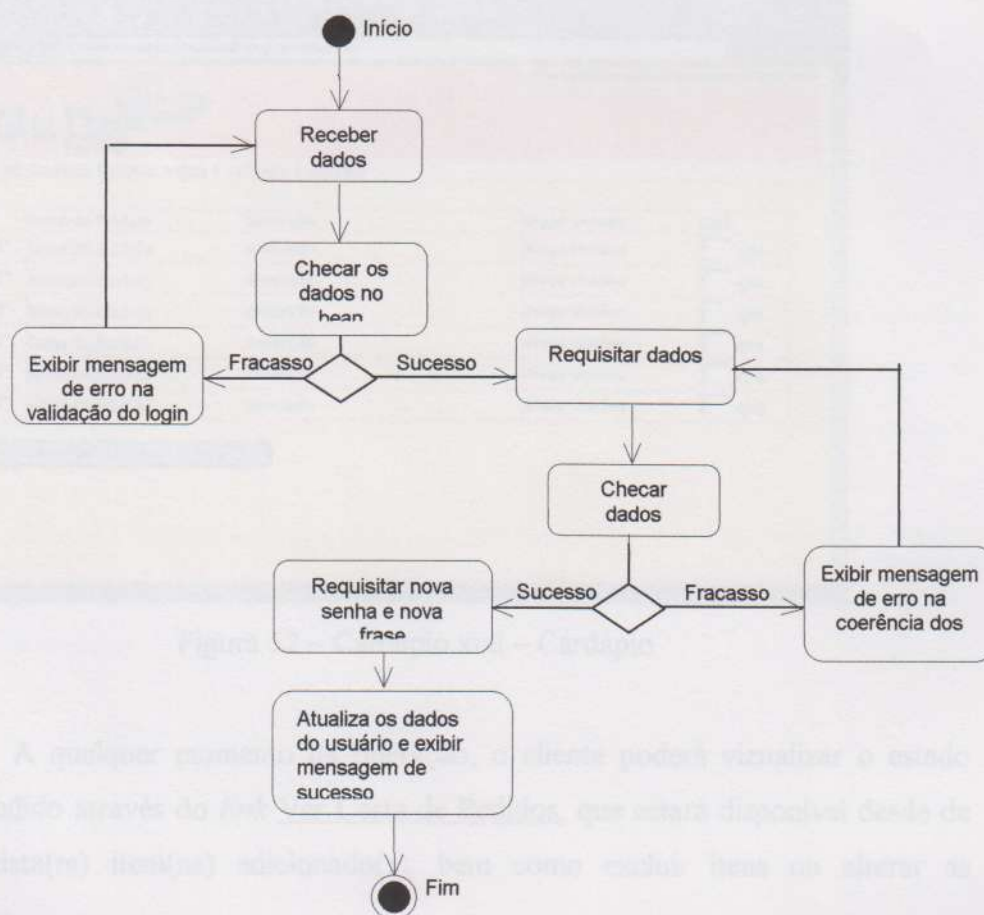


Figura 51 – Diagrama de Atividades Alterar Senha Esquecida: ServletChecaLogin, ServletChecaDados e ServletSenhaEsque

3.2.3 Ver Cardápio

Ao clicar, no link Ver Cardápio o sistema gera uma página .xml contendo o cardápio, que é composto de uma tabela com os campos: nome do produto, descrição, preço unitário e quantidade.

Se desejar visualizar o item, o cliente poderá clicar no nome do produto e o sistema exibe uma imagem do item desejado.

O cliente pode proceder com o pedido clicando na caixa de marcação do item, informando a quantidade desejada e, em seguida, clicando no botão Incluir Item(ns) Selecionado(s).

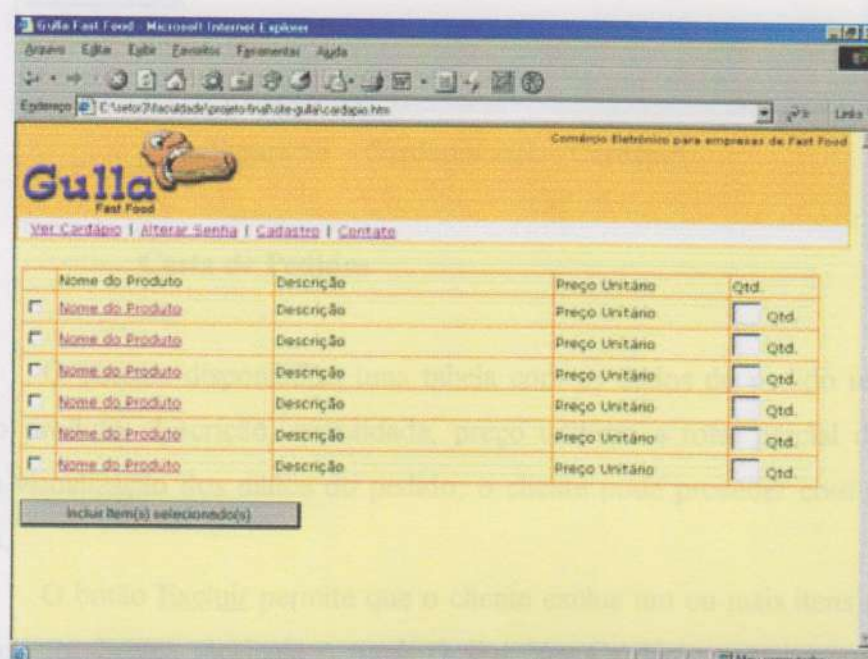


Figura 52 – Cardapio.xml – Cardápio

A qualquer momento da operação, o cliente poderá visualizar o estado atual do pedido através do link Ver Cesta de Pedidos, que estará disponível desde de que já exista(m) item(ns) adicionado(s), bem como excluir itens ou alterar as quantidades.

O pedido poderá ser concluído através do link Finalizar Pedido, que estará disponível desde que pelo menos um item tenha sido incluso no carrinho.

Estando na tela de Finalizar Pedido, o sistema requisita login, R.G e Senha, contendo também um link direcionando para Efetuar Cadastro caso o cliente não possua. Ocorrendo posteriormente a seleção da forma de pagamento.

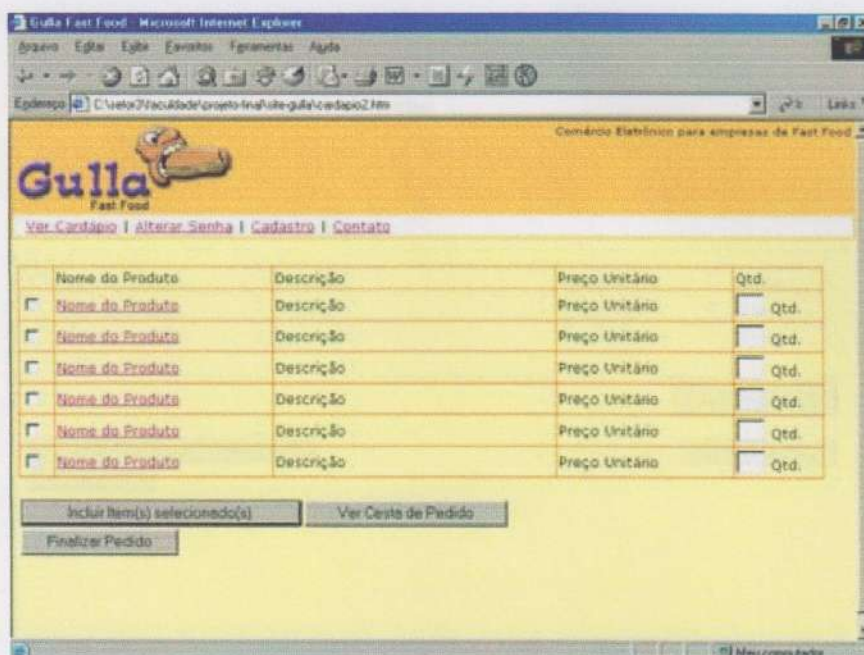


Figura 53 – Cardapio.xml – Cardápio

– Cesta de Pedidos

O sistema disponibiliza uma tabela com os dados do pedido nos campos nome do produto, descrição, quantidade, preço unitário e total parcial do item. A partir da visualização dos dados do pedido, o cliente pode proceder com a exclusão dos itens.

O botão Excluir permite que o cliente exclua um ou mais itens do pedido. Neste caso, o sistema recalcula o total. O link Ver Cardápio permite que o cliente retorne à tela anterior.



Figura 56 – Diagrama de Colaboração Excluir Item Cardápio

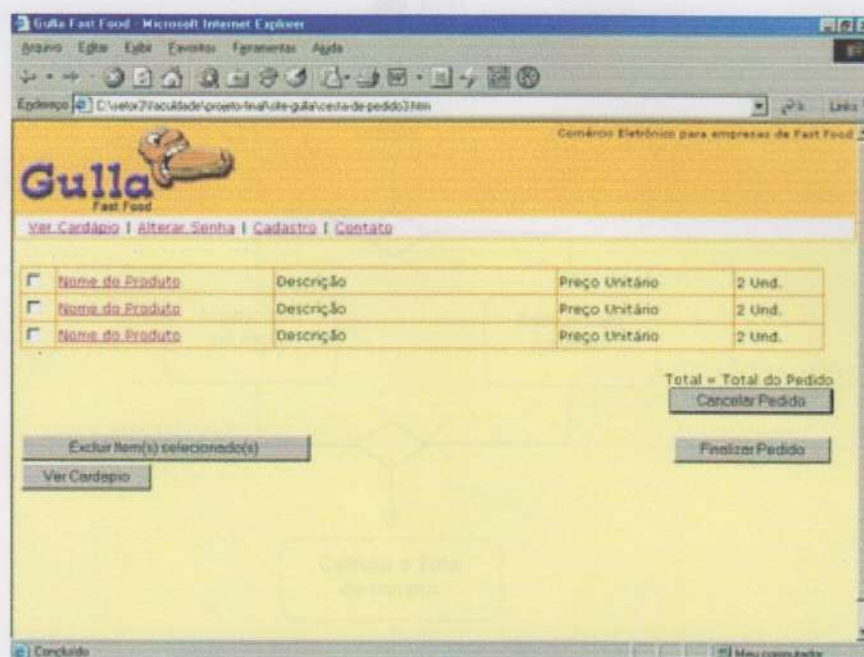


Figura 54 – cesta.jsp – Cesta de Pedido

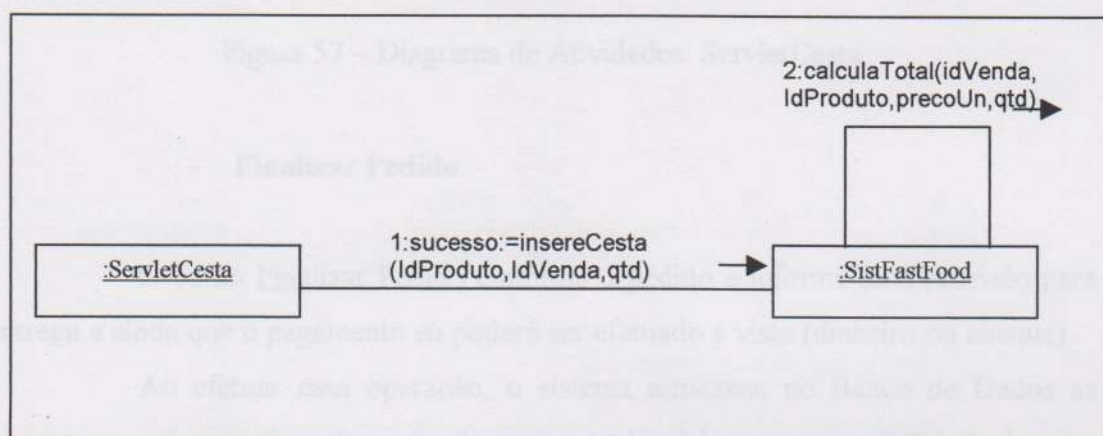


Figura 55 – Diagrama de Colaboração Inserir Item Cardápio

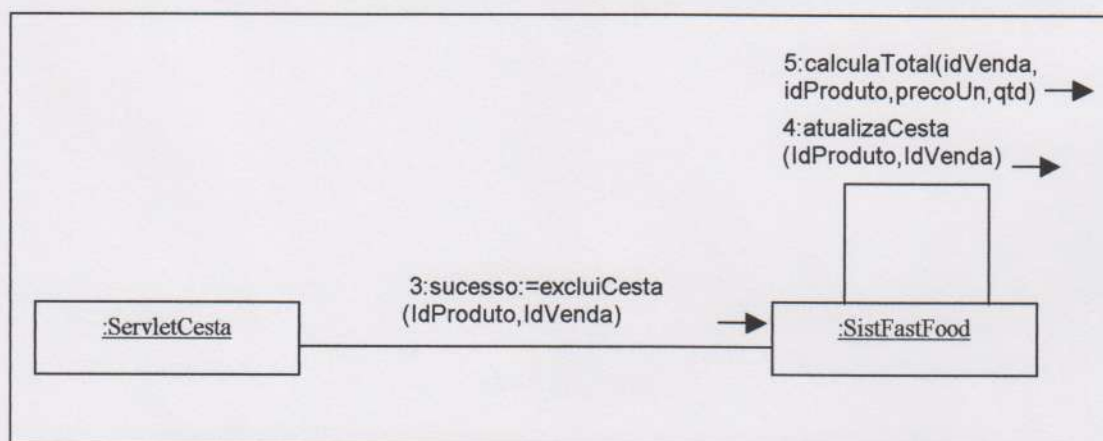


Figura 56 – Diagrama de Colaboração Excluir Item Cardápio

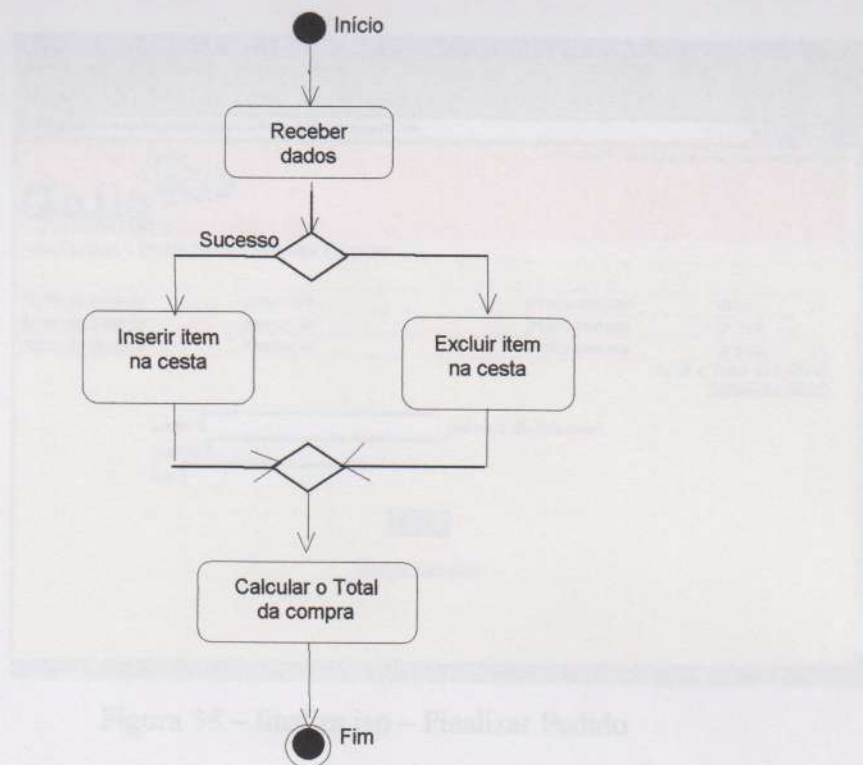


Figura 57 – Diagrama de Atividades: ServletCesta

– Finalizar Pedido

O botão Finalizar Pedido confirma o pedido e informa uma previsão para entrega e ainda que o pagamento só poderá ser efetuado à vista (dinheiro ou cheque).

Ao efetuar essa operação, o sistema armazena no Banco de Dados as compras que foram efetuadas, criando assim um histórico, com possibilidade de criar promoções personalizadas.

Figura 58 – Diagrama de Colaboração Finalizar Pedido

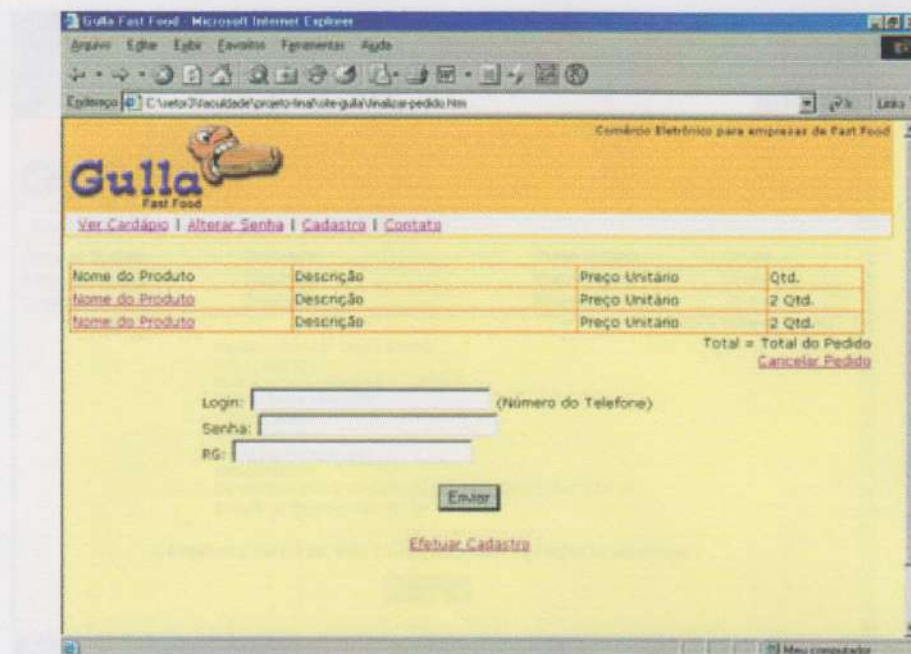


Figura 58 – finaliza.jsp – Finalizar Pedido

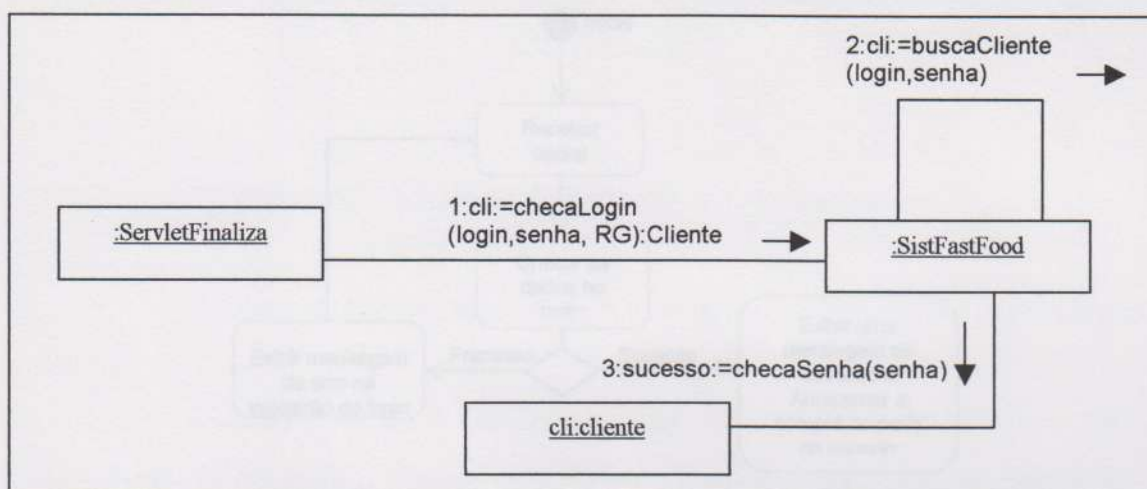


Figura 59 – Diagrama de Colaboração Finalizar Pedido

Figura 60 – Diagrama de Atividades: ServletFinaliza

3.3.4 Gerenciar Itens do Cardápio

O administrador tem acesso ao arquivo do cardápio do banco de dados, podendo realizar as operações de inclusão, alteração e exclusão de seus itens e servir as alterações para o servidor onde está alocado o web site.

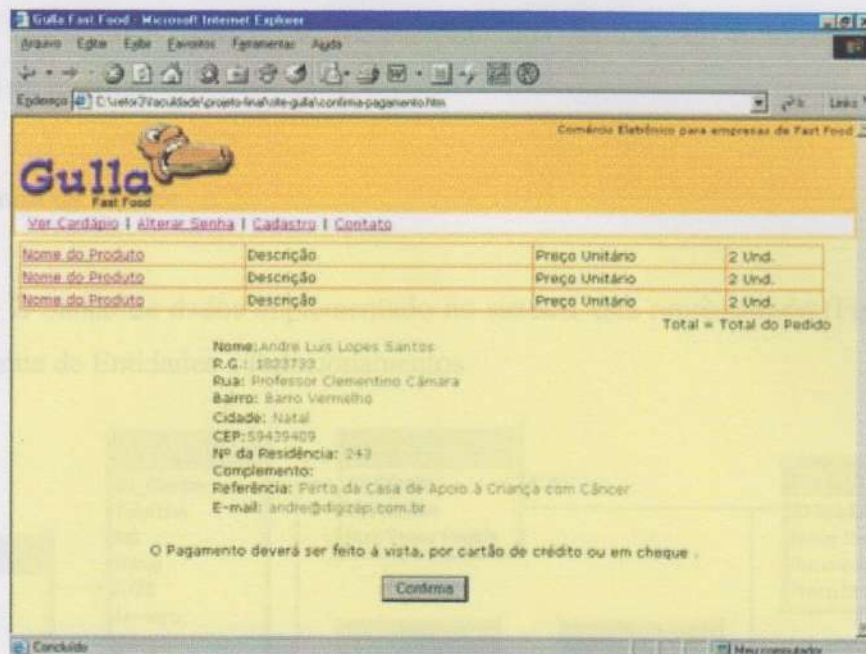


Figura 60 – finaliza.jsp – Finaliza Pedido

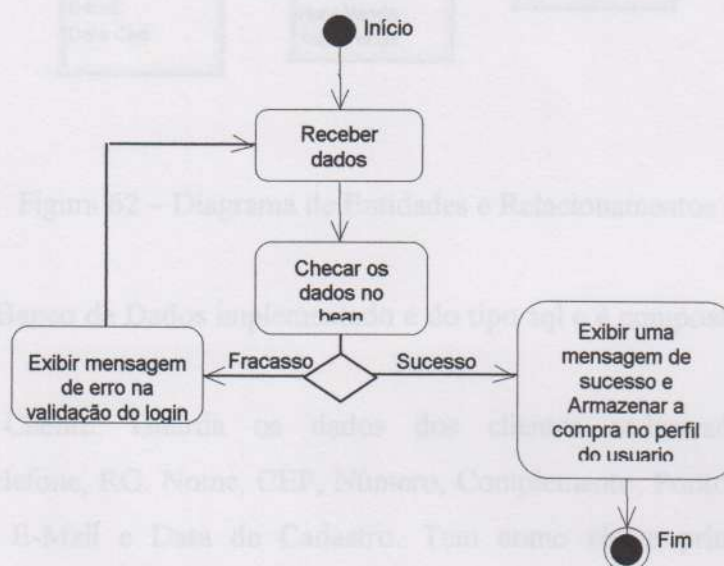


Figura 61 – Diagrama de Atividades: ServletFinaliza

3.2.4 Gerenciar Itens do Cardápio

O administrador tem acesso ao arquivo do cardápio no banco de dados, podendo realizar as operações de inclusão, alteração e exclusão de seus itens e enviar as alterações para o servidor onde está alocado o *web site*.

Logo após realizar a transmissão dos dados, as alterações estarão disponíveis na Internet.

3.3 Banco de Dados

O banco de dados implementado no sistema está representado (Figura 42) pelo Diagrama de Entidades e Relacionamentos.



Figura 62 – Diagrama de Entidades e Relacionamentos

O Banco de Dados implementado é do tipo sql e é composto das seguintes tabelas:

- **Cliente:** Guarda os dados dos clientes cadastrados, que são: ID_Cliente, Telefone, RG, Nome, CEP, Número, Complemento, Ponto de Referência, Senha, Frase, E-Mail e Data de Cadastro. Tem como chave primária o campo ID_Cliente e como chave estrangeira o campo CEP.
- **Venda:** Guarda os Dados de cada venda efetuada por cliente, que são: ID_Venda, ID_Cliente, Data Venda, Hora Venda e Total Venda. Tem como chave primária composta os campos ID_Venda e ID_Cliente.
- **Pedido:** Guarda os dados de cada produto contido em cada pedido, que são: ID_Venda, ID_Produto e Quantidade. Tem como chave primária composta os campos ID_Venda e ID_Produto.
- **Produto:** Guarda os dados de cada produto, que são: ID_Produto, Nome Item, Descrição e Preço Unitário. Tem como chave primária o campo ID_Produto.

4. **Conte** - Perfil: Guarda a quantidade de cada produto que já foi pedido por um determinado cliente. Possui os campos ID_Cliente, ID_Produto e Num Vezes Pedido. Tem como chave primária composta os campos ID_Cliente e ID_Produto.

um sistema de comércio eletrônico, na Internet, voltado para as empresas de setor alimentício, a partir de uma análise orientada a objetos, realizada em um trabalho anterior.

Propunha-se desenvolver um ambiente amigável, de fácil acesso e alta responsividade, habilitado a operar e praticar o e-commerce de uma maneira simples e intuitiva.

Visionou-se também a possibilidade de exploração dos serviços de e-commerce (negócios eletrônicos) pelas diversas lojas de fast-food, aumentando a qualidade no atendimento aos seus consumidores, bem como a redução de custos e complexidade dos sistemas atuais.

Para que fosse possível o desenvolvimento deste trabalho, utilizou-se a arquitetura de redes de computadores e Internet, baseadas nos princípios TCP/IP, bem como os conceitos de banco de dados relacionais.

O objetivo principal deste trabalho foi oferecer ao setor alimentício e aos seus consumidores um ambiente de implementação de comércio eletrônico.

Foi apresentada a descrição do sistema proposto, através de uma visão geral das funcionalidades do sistema, como também de uma descrição de interface gráfica de usuário de acordo com o modelo de desenvolvimento utilizado.

Dentre os benefícios imaginados para o projeto podem ser destacados os seguintes:

- Economia de tempo na execução dos pedidos;
- Facilidade de obtenção de publicidade para a empresa, com total visibilidade e disponibilidade 24 horas;
- Manutenção de um histórico com os pedidos dos clientes para elaboração de perfil e promoção personalizada;
- Baixo custo na obtenção e manutenção do equipamento.

Por fim, vale ressaltar ainda que, de forma simples e implementável, o sistema pode servir de base para o comércio eletrônico em qualquer área, desde que seja realizada alguma adaptação sobre o modelo dos dados utilizados.

4. Conclusões e Perspectivas

Este trabalho de graduação apresentou uma proposta de padronização de um sistema de comércio eletrônico, na Internet, voltado para as empresas do setor alimentício, a partir de uma análise orientada a objetos, realizada em um trabalho correlato.

Procurou-se desenvolver um ambiente amigável, de baixo custo e alta interatividade, habilitando o usuário a praticar o *e-commerce* de uma maneira simples e habitual.

Visou-se também a possibilidade da exploração dos serviços de e-business (negócios eletrônicos) pelas empresas locais de fast-food, aumentando a qualidade no atendimento aos seus consumidores, bem como a redução de custos e complexidade dos sistemas atuais.

Para que fosse possível o desenvolvimento deste trabalho, utilizou-se a arquitetura de redes de computadores e Internet, baseadas nos protocolos TCP/IP, bem como os conceitos de banco de dados relacional.

O objetivo principal deste trabalho foi oferecer ao setor alimentício e aos seus consumidores uma proposta viável de implementação de comércio eletrônico.

Foi apresentada a descrição da solução proposta, através de uma visão geral das funcionalidades do sistema, como também de uma descrição da interface escolhida e do banco de dados relacional utilizado.

Dentre os benefícios imaginados para o projeto podem ser destacados os seguintes:

- Economia de tempo na execução dos pedidos;
- Facilidade de obter/fornecer publicidade para a empresa, com custo reduzido e disponibilidade 24 horas;
- Manutenção de um histórico com os pedidos dos clientes para elaboração de perfis e promoções personalizadas;
- Baixo custo na obtenção e manutenção da ferramenta.

Por fim, vale ressaltar ainda que da forma como foi implementado, o sistema pode servir de base para o comércio eletrônico em qualquer área, já que não existe nenhuma restrição sobre o conteúdo dos dados utilizados.

Como perspectiva de trabalhos futuros, sugere-se a conexão do sistema com as empresas de cartão de crédito *on-line* para possibilitar uma outra forma de pagamento.

DATE, C. J. *Introdução a sistema de banco de dados*. Trad. Vandenberg Daniel de Souza. 7 ed. Rio de Janeiro: Campus, 2000.

DEITEL, H. M.; DEITEL, P. J. *Java: como programar*. Trad. Edson Furtadolewicz. 3 ed. Porto Alegre: Bookman, 2001.

PORTES, Debora; BOTTONI, Fernanda; VENTURA, Giuliano. *E-commerce: conheça os 100 nomes que dominam o B2C e o B2B e os investimentos on-line no Brasil*. Info-exame. São Paulo, ano 16, n. 182, p. 70, mai. 2001.

PORTES, Debora. *Software expões*. Info-exame. São Paulo, ano 16, n. 184, p. 103, jul. 2001.

FURGERI, Sergio. *Business-to-business: aprenda a desenvolver aplicações*. São Paulo: Érica, 2001.

GEARY, David M. *Dominando java server pages avançado*. Rio de Janeiro: Editora Ciência Moderna Ltda, 2002.

HTML. *Apostila de engenharia de redes*. Curso realizado em 23/11/1999.

JSP - JAVAServer Pages. Disponível em: <<http://www.sun.com>>. Acesso em 20/04/2002.

JSP - JAVAServer Pages. Disponível em: <<http://www.jspmail.com.br>>. Acesso em 01/05/2002.

LAKATOS, Eva Marie; MARCONI, Marconi de Andrade. *Fundamentos de metodologia científica*. 3 ed. São Paulo: Atlas, 1991.

NAKAMURA, Rodolfo. *E-Commerce na Internet*. São Paulo: Érica, 2001.

NJEMEYER, Patrick; KNUDSEN, Jonathan. *Aprendendo java*. Trad. Daniel Vieira. Rio de Janeiro: Campus, 2000.

OLIVEIRA, Silvio Luiz de. *Tratado de metodologia científica: projetos de pesquisa, TCC, monografias, dissertações e teses*. São Paulo: Pioneira, 1997.

SCHILD, Daniel; KOLB, Mark A. *Desenvolvimento na web com java server pages*. Rio de Janeiro: Ciência Moderna, 2000.

SOUKUP, Ron. *Desenvolvendo e administrando SQL server 6.5*. Trad. Vitor Alves; João Turneiro; Daniel Vieira; Premier Serviços Especializados. Rio de Janeiro: Campus, 1998.

REFERÊNCIAS BIBLIOGRÁFICAS

BOMFIM JUNIOR, Francisco Tarcizo. *A tecnologia java na internet*. São Paulo: Érica, 2002.

DATE, C. J.. *Introdução a sistema de banco de dados*. Trad. Vandenberg Dantas de Souza. 7 ed. Rio de Janeiro: Campus, 2000.

DEITEL, H. M.; DEITEL, P. J.. *Java, como programar*. Trad. Edson Furmankiewicz. 3 ed. Porto Alegre: Bookman, 2001.

FORTES, Débora; BOTTONI, Fernanda; VENTURA, Giuliano. *E-commerce - conheça os 100 nomes que dominam o B2C e o B2B e os investimentos on-line no Brasil*. Info-exame, São Paulo, ano 16, n. 182, p. 70, mai. 2001.

FORTES, Débora. *Software espiões*. Info-exame, São Paulo, ano 16, n.184, p. 108, jul. 2001.

FURGERI, Sergio. *Business-to-business: Aprenda a desenvolver aplicações*. São Paulo: Érica, 2001.

GEARY, David M.. *Dominando java server pages avançado*. Rio de Janeiro: Editora Ciência Moderna Ltda, 2002.

HTML. *Apostila de engenharia de redes*. Curso realizado em 23/11/1999.

JSP - JAVAServer Pages. Disponível em: <<http://www.sun.com>>. Acesso em 20/04/2002.

JSP - JAVAServer Pages. Disponível em: <<http://www.jspbrasil.com.br>>. Acesso em 01/05/2002.

LAKATOS, Eva Maria; MARCONI, Mariana de Andrade. *Fundamentos de metodologia científica*. 3 ed. São Paulo: Atlas, 1991.

NAKAMURA, Rodolfo. *E-Commerce na Internet*. São Paulo: Érica, 2001.

NIEMEYER, Patrick; KNUDSEN, Jonathan. *Aprendendo java*. Trad. Daniel Vieira. Rio de Janeiro: Campus, 2000.

OLIVEIRA, Silvio Luiz de. *Tratado de metodologia científica: projetos de pesquisas, TGI, TCC, monografias, dissertações e teses*. São Paulo: Pioneira, 1997.

SIELDF, Duanek; KOLB, Mark A.. *Desenvolvimento na web com java server pages*. Rio de Janeiro: Ciência Moderna, 2000.

SOUKUP, Ron. *Desenvolvendo o Microsoft SQL server 6.5*. Trad. Vitor Alves; João Tornelo; Daniel Vieira; Fremer Serviços Especiais. Rio de Janeiro: Campus, 1998.

TANENBAUM, Andrew S.. *Redes de computadores*. Trad. Insight Serviços de Informática. 3 ed. Rio de Janeiro: Campus, 1997.

SILBERSCHATZ, Abraham; KORTH, Henry; SUDARSHAN, S.. *Sistema de banco de dados*. Trad. Marília Pinheiro e Cláudio Canhette, São Paulo: Makron, 1999.

```
<!DOCTYPE HTML PUBLIC "-//W3C/DTD HTML 4.01 Transitional//EN">
<HTML>
<HEAD>
<TITLE>Sistema de Comercio Eletrônico - em JSP</TITLE>
</HEAD>
<BODY bgcolor="blue" text="white">
<CENTER>
<H1> Sistema de Comercio Eletrônico - em JSP</H1>
<FONT face="Arial">
<TABLE border="1" width="60%" bgcolor="black">
<TR>
<TH><FONT color="orange">Círculo do sistema</FONT></TH>
<TR>
<td><!-- Link para a função de cadastro do cliente -->
<TD><A href="cadastro.html">Cadastro Cliente</A>
</TABLE>
</FONT>
</CENTER>
</BODY>
</HTML>
```

Cadastro.html

```
<!DOCTYPE HTML PUBLIC "-//W3C/DTD HTML 4.01 Transitional//EN">
<HTML>
<HEAD>
<TITLE>Inserir um Cliente no Sistema</TITLE>
</HEAD>
<BODY bgcolor="blue" text="orange">
```

APÊNDICE A – CÓDIGO HTML

Index.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<HTML>
<HEAD>
<TITLE>Sistema de Comercio Eletrônico – em JSP</TITLE>
</HEAD>
<BODY bgcolor='blue' text='white'>
<CENTER>
<H1> Sistema de Comercio Eletrônico – em JSP</H1>
<FONT face='Arial'>
<TABLE border='1' width='60%' bgcolor='beige' >
<TR>
<TH> <FONT color='orange'>Opção do sistema:</FONT>
<TR>
<!--Link para a função de cadastro do cliente.-->
<TD> <A href='cadastro.html'>Cadastrar Cliente</A>
</TABLE>
</FONT>
</CENTER>
</BODY>
</HTML>
```

Cadastro.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<HTML>
<HEAD>
<TITLE>Insere um Cliente no Sistema</TITLE>
</HEAD>
<BODY bgcolor='blue' text='orange'>
```


APÊNDICE B – CÓDIGO JSP

ServletCadastro.jsp

```

<%@page contentType="text/html"%>
<%@page import="JSPBD1.*"%>
<html>
<head><title>Servlet para cadastro no Banco de Dados</title></head>
<body>
<jsp:useBean id="conexao" class="JSPBD1.BeanConexao" />
<CENTER><H2>
<% String login = request.getParameter("login");
    if (login == null) login = "test";
    String rg = request.getParameter("rg");
    if (rg == null) rg = "0"; %>
<% String sql = "INSERT INTO Clientes (login, rg) VALUES ("";
    sql = sql + login + "','" + rg + "')"; %>
<% int resultado = conexao.executeUpdate(sql);
    if (resultado == 1) { %>
        Inserção bem sucedida: Login = <%= login%> e RG = <%= rg%>
<% } else { %>
        Inserção ao banco não realizada!!
<% } %>
</H2></CENTER>
</body>
</html>

```

APÊNDICE C – CÓDIGO JAVA

Acesso.java

```
package JSPBD1;
import java.sql.*;
import java.util.*;

public class Acesso implements java.io.Serializable {
```

Método responsável pela transformação do resultado capturado no formulário do tipo inteiro para uma cadeia de caracteres.

```
public Vector transforma(ResultSet resultado) {
    ResultSetMetaData rsMetaDados;
```

```
    Vector tabela = new Vector();
```

```
    Vector linha = new Vector();
```

```
    try {
```

```
        rsMetaDados = resultado.getMetaData();
```

```
        while (resultado.next()) {
```

```
            linha = new Vector();
```

Laço que compara a posição dos caracteres para saber quantas posições irá ter a cadeia de caracteres.

```
            for (int i = 1; i <= rsMetaDados.getColumnCount(); i++) {
```

```
                switch (rsMetaDados.getColumnType(i)) {
```

Captura o tipo de dado passado pelo formulário e compara com o tipo de dado caractere, se for verdadeiro é adicionado um elemento na linha do vetor.

case Types.VARCHAR :

```
                    linha.addElement(resultado.getString(i));
```

```
                    break;
```

Captura o tipo de dado passado pelo formulário e compara com o tipo de dado inteiro, se for verdadeiro é adicionado um elemento na linha do vetor.

```
                case Types.INTEGER :
```

```
        linha.addElement(new Long(resultado.getLong(i)));
        break;
    }
}
Adiciona no objeto tabela um elemento linha.
tabela.addElement(linha);
}
Captura a exceção de erro em SQL que houver.
} catch (SQLException sqlex) {
    sqlex.printStackTrace();
}
Retorna o objeto tabela para o método que requisitou esse método.
return tabela;
}
}
```

<%@page contentType="text/html"%>