

**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**

SISTEMA DE ACOMPANHAMENTO DE PROJETOS VIA WEB

SISTEMA DE ACOMPANHAMENTO DE PROJETOS VIA WEB

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

Orientador:

Felipe Araújo Aladim de Araújo

Carla Moreira Aladim de Araújo

**NATAL/RN
2004**

CARLA MOREIRA ALADIM DE ARAÚJO
FACULDADE NATALENSE PARA O DESENVOLVIMENTO
DO RIO GRANDE DO NORTE
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

SISTEMA DE ACOMPANHAMENTO DE PROJETOS VIA WEB

SISTEMA DE ACOMPANHAMENTO DE PROJETOS VIA WEB

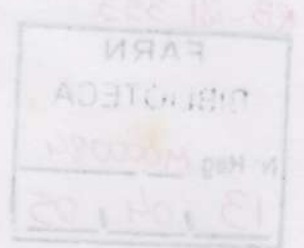
Carla Moreira Aladim de Araújo
FACULDADE NATALENSE PARA O DESENVOLVIMENTO
DO RIO GRANDE DO NORTE
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

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

Orientador:

Fellipe Araújo Aleixo, Msc.

NATAL/RN
2004



CARLA MOREIRA ALADIM DE ARAÚJO

SISTEMA DE ACOMPANHAMENTO DE PROJETOS VIA WEB

Catálogo na Publicação – Biblioteca da FARN
Setor de Processos Técnicos

Araújo, Carla Moreira Aladim de

Sistema de Acompanhamento de Projetos via Web / Carla Moreira Aladim de Araújo. __ Natal: FARN, 2004.

61p.

Orientador: Felliipe Araújo Aleixo (M.sc)

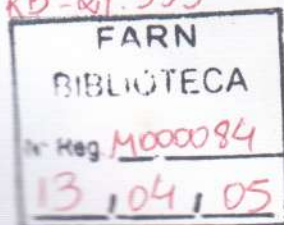
Monografia (Graduação em Sistemas de Informação) - Faculdade Natalense para o Desenvolvimento do Rio Grande do Norte.

1. Internet – monografia 2. Projetos via Web - monografia I. Título

RN/FARN/BC

CDU 004.738.5

RB-21.353



M
004.738.5
A6875
2004

NATALIAN
2004

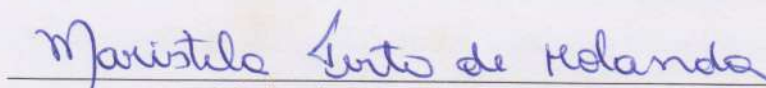
**LIGA DE ENSINO DO RIO GRANDE DO NORTE
FACULDADE NATALENSE PARA O DESENVOLVIMENTO
DO RIO GRANDE DO NORTE
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO**

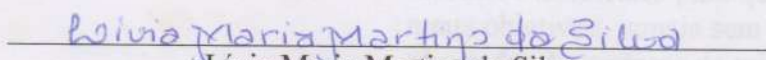
SISTEMA DE ACOMPANHAMENTO DE PROJETOS VIA WEB

Carla Moreira Aladim de Araújo

Monografia apresentada e aprovada em 11 de DEZEMBRO de 2004,
pela banca examinadora composta pelos seguintes membros


Felipe Araújo Aleixo
Orientador


Maristela Terto de Holanda
Membro Examinador


Livia Maria Martins da Silva
Membro Examinador

Natal, 11 de DEZEMBRO de 2004.

AGRADECIMENTOS

Este trabalho foi construído com o apoio de várias pessoas. Por isso, venho agradecer a todos com as mais sinceras palavras, esperando ter atendido os fins pretendidos com empenho, carinho e respeito às expectativas de todos que contribuíram para sua construção.

Aos meus professores, pelas ensinamentos ministrados e pelo estímulo aos desafios profissionais e acadêmicos que deles recebi.

Aos meus professores, que nos passaram todo o seu conhecimento, em especial ao Professor Felipe Araújo Almeida, que me ajudou bastante no decorrer desta monografia, disponibilizando sua atenção e seu tempo para me ajudar, perdendo suas horas de lazer, com os minutos indo para a faculdade para me ajudar e até mesmo perceber os erros e acertos deste trabalho, conduzindo-me a sempre fazer o melhor.

A Faculdade Natalense para o Desenvolvimento do Rio Grande do Norte, particularmente aos membros de suas Direções, Professores e Funcionários, pela presença e suporte que me ofereceram para a realização e conclusão desta oportunidade preciosa e valiosa que é a construção e aprimoramento do meu conhecimento.

Aos meus primos e amigos: Sílvia e Gabriela. Aos meus amigos Tiago, José Ronaldo, Vales, que com solidariedade, amizade e alegria, me ajudaram nas horas difíceis com uma palavra amiga e simples.

A minhas amigas que conquistaram o quarto junto comigo, mas não compartilharam Glenda, a amiga que apesar de não está comigo diariamente, me ajudaram muito.

A minha contábeis de orientação e amiga Pollyana Paz, por me emprestar as horas.

A toda a minha família, minhas irmãs Cynthia, Mirla, Marcela. Aos meus sobrinhos Lucas e João Pedro, por trazerem alegria para dentro de casa. Aos meus Pais Roberto e Maria das Graças por terem paciência comigo e ajudarem na conclusão desta monografia. A todos, que direta ou indiretamente, contribuíram na elaboração deste trabalho.

Dedico este trabalho a meus pais Maria das Graças e Joberto que contribuíram com o incentivo necessário para que eu alcançasse meus objetivos, jamais sem eles poderia ter resultado. Ao restante da minha família que de forma direta ou indireta, também ajudaram.

A Deus por me sustentar nas horas de fraqueza e me assegurar hoje à vitória, desse importante passo de minha vida.

AGRADECIMENTOS

Este trabalho foi construído com o apoio de várias pessoas. Por isso, venho agradecer a todos com as mais sinceras palavras, esperando ter atendido os fins pretendidos neste empenho conjunto e satisfeito as expectativas de todos que contribuíram para sua construção.

Assim, agradeço a todos os meus professores, pelos ensinamentos ministrados e pelo estímulo aos desafios profissionais e acadêmicos que deles recebi.

Aos meus professores, que nos passaram todo o seu conhecimento, em especial ao Professor Fellipe Araújo Aleixo, que me ajudou bastante no decorrer desta monografia, disponibilizando sua atenção e seu tempo para me ajudar, perdendo suas horas de lazer, como os sábados indo para a faculdade para me ajudar e este soube perceber os erros e acertos deste trabalho, conduzindo-me a sempre fazê-lo melhor.

À Faculdade Natalense para o Desenvolvimento do Rio Grande do Norte, particularmente nas pessoas de seus Diretores, Professores e Funcionários, pela presteza e suporte que me ofereceram para a realização e conclusão desta oportunidade produtiva e valorosa que é a construção e aprimoramento do meu conhecimento.

Aos meus primos e amigos Stenio e Gabriela. Aos meus amigos Tiago, José Renaldo, Stella, que com solidariedade, carinho e alegria, me ajudaram nas horas difíceis com uma palavra amiga e sincera.

As minhas amigas que começaram o curso junto comigo, mas não concluíram Glenda e Luciana que apesar de não está comigo diariamente, me ajudaram muito.

A minha companheira de orientação e amiga Polyana Paz, agradeço por me emprestar os livros.

A toda a minha família, minhas irmãs Cynthia, Márcia, Marcela. Aos meus sobrinhos Lucas e João Pedro, por trazerem alegria para dentro de casa. Aos meus Pais Joberto e Maria das Graças por terem paciência comigo e ajudarem na conclusão desta monografia, a todos, que direta ou indiretamente, contribuíram na elaboração deste trabalho. O meu mais profundo obrigada.

RESUMO

Com os avanços que a computação e as telecomunicações tem alcançado nos dias atuais, os benefícios decorrentes para a sociedade de uma forma geral são incontestáveis. Esses avanços vêm tornando as redes de computadores cada vez mais confiáveis e utilizadas por toda a sociedade. Com o crescimento da Internet, rede de alcance mundial que está disponível continuamente, a cada dia surge novos serviços. Um dos maiores usos da Internet até hoje, é a troca de informações. Com o aumento destas trocas de informações, começaram a surgir os sites de busca e de depósito de informações, que são sites que armazenam materiais e possibilitam buscas por documentos específicos. Sistemas como estes estão sendo cada vez mais utilizados pelas instituições em geral e pela sociedade. Este sistema em questão será como um site onde se troca as informações, Onde irá ser utilizado, como, por exemplo, na Faculdade Natalense para o desenvolvimento do Rio Grande do Norte, Tal sistema tem como finalidade ajudar no relacionamento entre um Orientador e seus Orientandos, podendo melhorar a relação entre eles.

Palavras - Chave: Internet, Projetos via Web.

ABSTRACT

Along with the advancements that computation and telecommunication have reached actually, there are benefits occurred from those advancements for the whole society, in a general way, that are undeniable. These advancements are making computer webs even more trustable and useful for all the users. As internet (world wide reach web that is available incessantly) improves, new services come up everyday. One of the biggest deals of the internet until today is the information exchange. With the increase of these exchanges, research web sites and information replacements, web sites that store materials and make possible searches for an specific document, appeared. Systems like these are being used by establishments in general and by the society even more each time.

This system will be like a web site where information exchanging will happen and will be used as, for example, by FARN. This system's purpose is to make the relationship between the person who orientates and the person orientated easier and better.

Key - words: Internet, Web Way Projects.

SUMÁRIO

1. INTRODUÇÃO.....	11
1.1 OBJETIVOS.....	12
1.1.1 Objetivos Gerais.....	12
1.1.2 Objetivos Específicos	12
1.2 JUSTIFICATIVA.....	13
2. REFERENCIAL TEORICO.....	14
2.1 INTERNET.....	16
2.2 HTML.....	15
2.2.1 Tabelas HTML.....	17
2.2.2 Formulários HTML.....	19
2.3 UML (LINGUAGEM DE MODELAGEM UNIFICADA).....	20
2.3.1 Diagrama de Casos de uso.....	20
2.3.2 Diagrama de Classes.....	23
2.3.3 Diagrama de Sequência.....	25
2.4 JAVA.....	26
2.4.1 JDBC (Java Database Connectivity).....	27
2.4.2 JSP (Java Server Pages).....	29
2.4.3 Servlets.....	30
2.4.4 Javabeans.....	30
2.5 BANCO DE DADOS.....	33
2.5.1 Sql (Structured Query Language).....	31
3. ANÁLISE DO SISTEMA DE ACOMPANHAMENTO.....	33
DE PROJETOS VIA WEB	
3.1 DOCUMENTO DE REQUISITOS.....	33
3.2 DIAGRAMA DE CASOS DE USO.....	35
3.3 CASOS DE USO DE ALTO NÍVEL.....	36
3.4 CASOS DE USO EXPANDIDO E DIAGRAMA.....	39
DE SEQUÊNCIA	
3.5 DIAGRAMA DE CLASSE.....	43
4. IMPLEMENTAÇÃO DO SISTEMA DE ACOMPANHAMENTO.....	45
DE PROJETOS VIA WEB	
5. CONCLUSÃO.....	51
REFERÊNCIAS BIBLIOGRAFICAS.....	52
BIBLIOGRAFIA CONSULTADA.....	53
ANEXO I - CÓDIGOS JSP.....	55
ANEXO II - CÓDIGOS JAVA.....	59
Figura 36- Diagrama de Classes do sistema.....	44
Figura 37- Tela Inicial do sistema.....	43
Figura 38- Tela do Administrador.....	45
Figura 39- Tela do Orientador.....	47
Figura 40- Tela para cadastrar projeto no sistema.....	48

LISTA DE FIGURAS

Figura 1-	Estrutura básica de um documento HTML.....	15
Figura 2-	Exemplo de como é exibida uma tabela em um browser.....	16
Figura 3-	Exemplo do código de uma tabela em HTML.....	16
Figura 4-	Estrutura básica de um Formulário em HTML.....	17
Figura 5-	Exemplo do Tag Input Type.....	17
Figura 6-	Estrutura básica do diagrama de casos de uso.....	19
Figura 7-	Estrutura de um caso de uso de alto nível.....	19
Figura 8-	Explicação de um caso de uso expandido.....	20
Figura 9-	Exemplo de um Caso de uso expandido.....	21
Figura 10-	Estrutura de uma classe.....	22
Figura 11-	Exemplo de generalização.....	22
Figura 12-	Classificação de multiplicidades.....	23
Figura 13-	Exemplo de um diagrama de classes.....	23
Figura 14-	Exemplo de um diagrama de sequência.....	24
Figura 15-	Estrutura do JDBC.....	27
Figura 16-	Exemplo de uma JSP.....	29
Figura 17-	Funções do sistema.....	33
Figura 18-	Atributos do sistema.....	33
Figura 19-	Diagrama de casos de uso do sistema.....	34
Figura 20-	Caso de uso de alto nível – Ler descrição de uma tarefa.....	35
Figura 21-	Caso de uso de alto nível – Cadastrar Projeto.....	35
Figura 22-	Caso de uso de alto nível – Indicar finalização de uma tarefa.....	36
Figura 23-	Caso de uso de alto nível – Cancelar tarefa.....	36
Figura 24-	Caso de uso de alto nível – Indicar dúvida.....	36
Figura 25-	Caso de uso de alto nível – Alterar descrição de uma tarefa.....	37
Figura 26-	Caso de uso de alto nível – Verificar tarefas em atraso.....	37
Figura 27-	Caso de uso de alto nível – Verificar tarefas finalizadas.....	38
Figura 28-	Caso de uso de alto nível – Atribuir tarefa.....	38
Figura 29-	Caso de uso expandido – cadastrar projeto.....	39
Figura 30-	Caso de uso expandido - Indicar finalização de uma tarefa.....	40
Figura 31-	Caso de uso Expandido – Indicar dúvida.....	41
Figura 32-	Diagrama de sequência – Indicar finalização de uma tarefa.....	41
Figura 33-	Diagrama de sequência – Cadastrar Projeto.....	42
Figura 34-	Diagrama de sequência – Indicar dúvida.....	42
Figura 35-	Diagrama de sequência – Atribuir tarefa.....	43
Figura 36-	Diagrama de Classes do Sistema.....	44
Figura 37-	Tela inicial do sistema.....	45
Figura 38-	Tela do Administrador.....	46
Figura 39-	Tela do Orientador.....	47
Figura 40-	Tela para cadastrar projeto no sistema.....	48

1. INTRODUÇÃO

A troca de informações é uma das maiores utilidades da Internet até hoje. Tais avanços fazem com que as redes de computadores se tornem mais confiáveis e utilizadas por toda a sociedade. Pois aumenta a confiabilidade do sistema, economiza dinheiro, e outra vantagem oferecida pela rede de computadores é a escalabilidade. Com o aumento da aceitação da Internet, surgem diversos serviços que visam suprir as necessidades das pessoas que procuram mais comodidade, informações e rapidez. Este trabalho trata da implementação de um sistema de acompanhamento de projetos, o qual ajudará na relação de um orientador com seus orientados. Tem como intenção ajudar os orientadores que possuem um grande número de orientandos, e resolver problemas como a dificuldade de comunicação individual, além da flexibilização dos horários destinados às orientações, pois o sistema funcionará on-line na Internet. Outra vantagem do sistema é controlar as tarefas e prazos que cada participante do projeto têm que cumprir. O sistema possibilita até mesmo que um orientando possa ser orientado por mais de um orientador.

1.1 OBJETIVOS

1.1.1 Objetivos Gerais

Este trabalho trata da implementação de um protótipo de um sistema de acompanhamento de projetos através da plataforma web.

1.1.2 Objetivos Específicos

- Ajudar um orientador a gerenciar os seus orientandos;
- Facilitar o acesso Orientador / Orientando, através da Internet;
- Facilitará as atividades realizadas pelos professores e alunos que estarão sendo beneficiados no sentido de comodidade e rapidez de adquirir as informações acadêmicas.

1. REFERENCIAL TEÓRICO

1.2 JUSTIFICATIVA

1.2.1 INTERNET

O produto final deste projeto irá proporcionar uma maior interação entre o orientador e seus orientandos. Comparando este sistema com o sistema de correio eletrônico (*e-mail*), percebemos que se fosse utilizado o *e-mail* para controle das tarefas, isto obrigaria ao orientador ter que saber o endereço eletrônico de cada um dos orientandos, para a perfeita distribuição das tarefas, bem como para identificar qual o aluno que enviou cada *e-mail*. Caso seja utilizado o sistema proposto, após o cadastramento do projeto e dos participantes (orientadores e orientandos): o orientador terá controle sobre as tarefas sendo desenvolvidas e seus executores, bem como os prazos; e os orientandos terão acesso às suas atribuições, podendo dar baixa no sistema, quando estiverem concluídas. Haverá uma lista das tarefas finalizadas e pendentes, divididas por ordem crescente da data prevista para finalização. Quando a tarefa for finalizada, já vai automaticamente para a lista das tarefas concluídas. Outra vantagem do sistema é controlar as tarefas e prazos que cada aluno têm que cumprir.

Uma página Web dispõe de recursos diferenciados. O caminho de uma URL é a sequência completa e padronizada o nome de arquivo da página Web. O protocolo e o host são sempre necessários, mas o caminho e o nome de arquivo não são necessários para todas as páginas Web.

Servidor é o computador que possui o arquivo desejado, localizado em qualquer parte do mundo e ligado à Internet.

O software de Cliente utilizado para acessar a *www* é denominado navegador (browser) este programa que permite que o usuário navegue pela Internet e visualize as páginas Web.

Para navegar na Internet é preciso estar conectado e ter um navegador web. Existem vários navegadores, mas o que você irá selecionar dependerá em parte do tipo de acesso escolhido. Existem vários como, por exemplo, o Internet Explorer, o Netscape Communicator, Altavista Online, etc.

O serviço *www* oferece um ambiente de hipertexto para o acesso à Internet. Ao contrário das outras ferramentas de rede, ele apresenta imagens, reproduz sons e até mesmo filmes, desde que você tenha o equipamento necessário. Os criadores de páginas da WWW estão se aproveitando das vantagens da multimídia para projetar materiais interativos.

A *www* oferece uma visão de hipertexto da Internet, não de texto e Internet, mas de grande parte do material que ela contém. Ela oferece (página na *www*) sua própria estrutura, como, por exemplo, pode ser colocada e não há nada que chamo mais atenção dos usuários. Ela também oferece como outras páginas na web, servidores Gopher, sites FTP etc.

2. REFERENCIAL TEÓRICO

2.1 INTERNET

A Internet é uma gigantesca rede de computadores, à qual qualquer pessoa pode conectar seu computador.

Segundo Alcântara (1997) "World Wide Web é o mais poderoso sistema de informação da Internet e o que mais tem se desenvolvido".

Segundo Comer (1999) "O www integra recursos multimídia e hipertexto em uma interface gráfica bastante interativa".

Uma página web é uma combinação de texto e tags de HTML (Hyper Text Markup Language) que formatam esses textos.

Uma URL (Uniform Resource Locator), especifica o endereço de cada página Web na Internet. Uma URL tem três partes: protocolo, host e caminho. O nome de domínio do host Internet especifica em que a página Web reside o recurso referenciado. O caminho de uma URL é a sequência completa e possivelmente o nome do arquivo da página Web. O protocolo e o host são sempre requeridos, mas o caminho e o nome do arquivo não são requeridos para todas as páginas Web.

Servidor é o computador que possui o arquivo desejado, localizado em qualquer parte do mundo e ligado à Internet.

O software de Cliente utilizado para acessar a www é denominado navegador (Browser) este programa que permite que o usuário navegue pela Internet e visualize as páginas Web.

Para navegar na Internet é preciso estar conectado e ter um navegador web. Existem vários disponíveis, mas o que você irá selecionar dependerá em parte do tipo de acesso escolhido. Existem vários como, por exemplo: o Internet Explorer, o Netscape Communicator, América Online, etc...

O Serviço www oferece um ambiente de hipermídia para o acesso à Internet. Ao contrário das outras ferramentas da rede, ela apresenta imagens, reproduz som e até exibe filmes, desde que você tenha o equipamento necessário. Os criadores de páginas da WWW estão se aproveitando dos serviços de multimídia para projetar materiais interativos.

A www oferece uma visão de hipertexto da Internet, não de toda a Internet, mas de grande parte do material que ela contém. Ela oferece (página na www) sua própria atratividade, como, por exemplo, pode ser colorida e isso faz com que chame mais atenção dos usuários. Ela contém vínculos com outras páginas da web, servidores Gopher, sites FTP etc.

2.2 HTML (Hyper Text Markup Language)

Segundo Alcântara (1997) “É a linguagem padrão adotada pela Web desde 1990 para criar e reconhecer documentos. Todos os documentos hipertexto da Web são escritos nessa linguagem. A linguagem HTML é uma linguagem de “marcação”. Uma linguagem de marcação se caracteriza por dividir o texto em vários pedaços funcionais, que são identificados por elementos de marcação ou tags”.

Outro importante conceito na Web é a home page. Uma home page é um hipertexto projetado para ser a página inicial, de uma coleção de páginas conectadas, a ser vista pelo usuário. Esse conjunto de páginas é denominado site. Uma home page deve ter como objetivo básico o direcionamento de espectadores para outros documentos relacionados.

```
<HTML>
  <HEAD> cabeçalho <TITLE> título </TITLE></HEAD>
  <BODY>
    Corpo do programa
  </BODY>
</HTML>
```

Figura 1. Estrutura básica de um documento HTML.

O tag HTML determina extensão do documento HTML.

O tag HEAD é o cabeçalho, área onde são colocadas informações adicionais sobre a página.

O tag TITLE define o título da página.

O tag BODY é o corpo, onde define o conteúdo a ser exibido.

2.2.1 Tabelas HTML

As tabelas servem para melhor visualizar, e organizar informações. Elas são iniciadas como os tags <TABLE> ... </TABLE>. Estes tags são colocados dentro do tag BODY.

- Especifica uma linha da tabela:

```
<TR>...</TR>
```

- Especifica as células de dados de uma tabela:

```
<TD>...</TD>
```

- Especifica uma espessura para a borda da tabela, que pode se atribuir um valor para este, se não for atribuído nenhum valor ou o valor for igual a zero, a tabela ficara sem borda, exatamente como o caso padrão:

```
<TABLE BORDER = "0">
```

- Especifica a largura da tabela em pixels ou em valores percentuais:

```
<TABLE WIDTH = "80%">
```

LIN 1 COL 1	LIN1 COL 2
LIN 2 COL 1	LIN 2 COL 2
LIN 3 COL 1	LIN 3 COL 2

Figura 2: Exemplo de como é exibida uma tabela em um browser.

Código fonte para esta tabela:

```
<HTML>
<HEAD><TITLE> TABELA </TITLE> </HEAD>
<BODY>
<TABLE BORDER WIDTH = "30%">
  <TR>
    <TD> LIN 1 COL 1</TD>
    <TD> LIN 1 COL 2</TD>
  </TR>
  <TR>
    <TD> LIN 2 COL 1</TD>
    <TD> LIN 2 COL 2</TD>
  </TR>
  <TR>
    <TD> LIN 3 COL 1</TD>
    <TD> LIN 3 COL 2</TD>
  </TR>
</TABLE>
</BODY>
</HTML>
```

Figura 3. Exemplo do código de uma tabela em HTML.

2.2.2 Formulários HTML

Os formulários são uma forma de interação com os usuários que acessam uma página web

Os tags que definem um formulário são:

`<FORM> ... </FORM>`

```
<FORM ACTION = "url" METHOD = "GET/POST">
```

Campos de entrada de dados

```
</FORM>
```

Figura 4. Estrutura básica de um formulário.

O atributo METHOD define o método de passagem dos dados para o "script".

O atributo ACTION especifica a url para a qual o conteúdo do formulário será mandado.

O método GET inclui as informações no final da url.

O método POST envia todas as informações sem que possam ser vistas no corpo da mensagem de solicitação.

```
<INPUT TYPE = "Tipo do campo de entrada de dados" NAME = "nome" >
```

Figura 5. Exemplo do Tag Input Type.

Tipos de entrada de dados:

- a) TEXT: Entradas de texto, usado quando se deseja inserir uma string do usuário como nome, endereço, etc.
- b) PASSWORD: É como o tipo TEXT, mas o texto digitado aparecerá em asterisco já que se trata de uma senha.
- c) CHECKBOX: Quando o usuário deve escolher uma ou mais resposta de um conjunto de alternativas.
- d) RADIO: Quando o usuário deve escolher apenas uma resposta de um conjunto de alternativas.
- e) SUBMIT: Possibilitam o envio dos dados para o script que vai tratá-los.
- f) RESET: Quando o botão é clicado, ele automaticamente limpa todos os campos já preenchidos no formulário.
- g) HIDDEN: pode guardar informações no formulário, sem que o usuário veja.

2.3 UML (LINGUAGEM DE MODELAGEM UNIFICADA)

Segundo Fowler (2000), “UML é a sucessora da onda de métodos de análise e projeto orientado a objetos que surgiu no final dos anos oitenta e no início dos anos noventa. Mais especificamente ela unifica os métodos de Booch, Rumbaugh e Jacobson, mas seu alcance é bem maior”.

2.3.1 Diagrama de Casos de uso

Um diagrama de Casos de uso de um sistema é uma das ferramentas essenciais na captura de requisitos e no planejamento da execução para criar um projeto iterativo.

Um caso de uso é um documento narrativo que descreve a sequência de eventos de um ator (um agente externo) que usa um sistema para completar um processo. Eles são histórias ou casos de utilização de um sistema. Casos de uso não são exatamente uma especificação de requisitos ou especificação funcional, mas ilustram e implicam requisitos na história que eles contam.

Casos de uso é uma técnica de modelagem usada para descrever o que um novo sistema deve fazer. Ele é construído através de um processo iterativo no qual as discussões entre o cliente e os desenvolvedores do sistema conduzem a uma especificação do sistema da qual todos estão de acordo. Os casos de uso são interações entre os atores e o sistema. Temos então: ações do ator e respostas do sistema sendo que os atores sempre iniciam a ação.

Um diagrama de Casos de uso ilustra um conjunto de casos de uso para um sistema, os atores e a relação entre eles.

Os casos de uso têm por objetivo:

- Descrever os requisitos funcionais do sistema.
- Fornecer uma descrição clara e consistente do que o sistema deve fazer.
- Permitir descobrir os requisitos funcionais das classes e operações do sistema (Casos de uso não são requisitos).

Os atores desempenham os casos de uso. Um único ator pode desempenhar muitos casos de uso; um Caso de uso pode ter vários atores desempenhando-o.

Os atores não precisam ser humanos, ele pode ser um outro sistema que interage de alguma forma com o sistema atual.

Um Ator é uma entidade externa ao sistema que, de alguma maneira, participa da história do caso de uso. Um ator, tipicamente, estimula o sistema com eventos de entrada ou recebe algo do

mesmo. Os atores são representados pelo papel que eles desempenham no caso de uso, tais como Usuário, Caixa e assim por diante. É desejável começar com maiúsculas o nome dos atores no texto do caso de uso para facilidade de identificação.

Geralmente os atores são descritos, utilizando-se:

- Nome do ator
- Descrição de seu papel no sistema

Na UML, o modelo de Casos de uso consiste de diagramas de Casos de uso que mostram os atores, os casos de uso e seus relacionamentos. Os elementos gráficos que representam atores, casos de uso e sistema são mostrados abaixo:

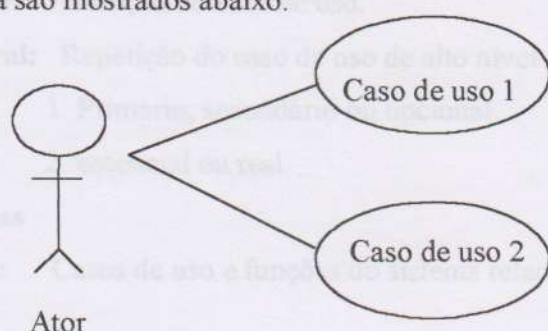


Figura 6. Estrutura básica do diagrama de casos de uso.

Casos de uso primários representam processos comuns principais.

Casos de uso secundários representam processos menos importantes ou raros.

Casos de uso opcionais representam processos que podem não ser considerados.

Casos de uso de alto nível

Um caso de uso de alto nível descreve um processo de forma muito breve. É útil criar este tipo de caso de uso durante a especificação inicial de requisitos e determinação do escopo inicial do projeto, de forma a compreender rapidamente o grau de complexidade e a funcionalidade de um sistema.

Caso de Uso:	Nome do caso de uso.
Atores:	Lista de atores (agentes externos), indicando quem inicia o caso de uso.
Tipo:	1. Primário, secundário ou opcional (a ser discutido).
Descrição:	Função do caso de uso.

Figura 7: Estrutura de um caso de uso de alto nível.

Casos de uso Expandido

Um caso de uso expandido mostra mais detalhes do que um de alto nível; eles são úteis para obter uma compreensão mais profunda dos processos e requisitos. Eles são freqüentemente executados em um estilo "conversacional" entre os atores e o sistema.

A parte superior da forma expandida é informação de resumo.

Caso de uso: Nome do caso de uso.

Atores: Lista de atores (agentes externos), indicando quem inicia o caso de uso.

Finalidade: Intenção do caso de uso.

Visão Geral: Repetição do caso de uso de alto nível ou algum sumário similar.

Tipo: 1. Primário, secundário ou opcional.
2. essencial ou real.

Referências

Cruzadas: Casos de uso e funções do sistema relacionadas.

A sequência *Típica de Eventos* é o coração do formato expandido, ela descreve detalhes a conversação da interação entre os atores e o sistema.

Sequência Típica de Eventos

Ações do Ator

Ações numeradas dos Atores

Resposta do Sistema

Descrições numeradas das respostas do sistema

A sequência alternativa de eventos descreve alternativas importantes, ou exceções, que podem surgir em relação à sequência típica.

Sequências alternativas

- Alternativas que podem surgir por número de linha. Descrições de exceções.

Figura 8. Explicação de um caso de uso Expandido.

Caso de uso: Comprar Itens com dinheiro

Atores: Cliente (iniciador), Caixa

Finalidade: Capturar uma venda e seu pagamento em dinheiro.

Visão Geral: Um cliente chega a um ponto de pagamento, com vários itens que deseja comprar. O caixa registra os itens da compra e recebe um pagamento em dinheiro. Ao término, o cliente sai com os itens.

Tipo: primário e essencial

Referências Cruzadas: Funções R1.1, R1.2, R1.3, R1.7, R1.9, R2.1

Sequência Típica de Eventos

Resposta do Sistema

- | | |
|---|--|
| <p>1. Este caso de uso começa quando um cliente chega a um ponto de pagamento com vários itens que deseja comprar.</p> <p>2. O caixa registra o identificador de cada item. Se houver mais de um exemplar do mesmo item, o caixa também pode entrar com a Quantidade.</p> <p>4. Ao término da entrada de itens, o caixa indica ao Post que a entrada de itens está completa.</p> <p>6. O caixa informa ao cliente o total.</p> <p>7. O cliente dá um pagamento em dinheiro— “o valor fornecido” — possivelmente maior que o total da venda.</p> <p>8. O caixa registra o montante de dinheiro recebido</p> <p>10. O caixa deposita o dinheiro recebido e retira o troco devido. O caixa dá ao cliente o troco e o recibo impresso.</p> <p>12. O cliente sai com os itens comprados.</p> | <p>3. Determina o preço do item e acrescenta a informação sobre o item à transação corrente de vendas. A descrição e o preço do item corrente são exibidos.</p> <p>5. Calcula e apresenta o total da venda.</p> <p>9. Exibe o valor do troco a ser devolvido ao cliente. Gera um recibo.</p> <p>11. Registra a venda completada.</p> |
|---|--|

Figura 9: Exemplo de um caso de uso expandido.

2.3.2 Diagrama de Classes

Descreve classe de objetos que farão parte do sistema e os vários tipos de relacionamento que existem entre elas, eles também mostram atributos e métodos de uma classe. Os diagramas de Classes são à base de quase todas as metodologias Orientadas a Objetos, portanto irá ser muito utilizada no decorrer do processo.

Uma interface é uma interconexão entre dois equipamentos que possuem diferentes funções e que não se poderiam conectar.

Em um modelo de especificação, generalização significa que a interface do subtipo deve incluir todos os elementos da interface do supertipo.

A Generalização, na perspectiva de implementação é associada à Herança nas linguagens de Programação. A subclasse herda todos os métodos e campos da superclasse e pode sobrescrever métodos herdados.

CLASSE

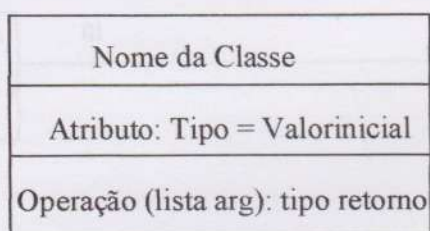


Figura 10. Estrutura de uma classe.

GENERALIZAÇÃO

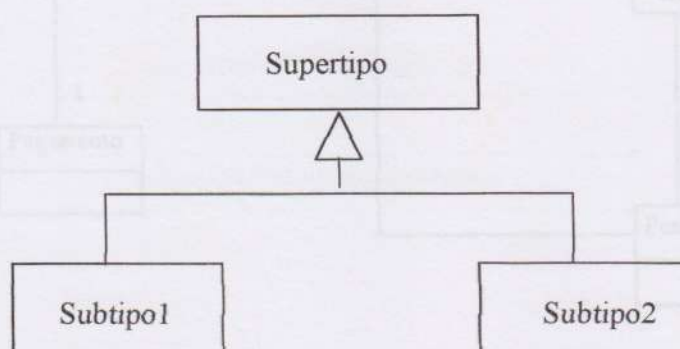


Figura 11. Exemplo de generalização.

MULTIPLICIDADES

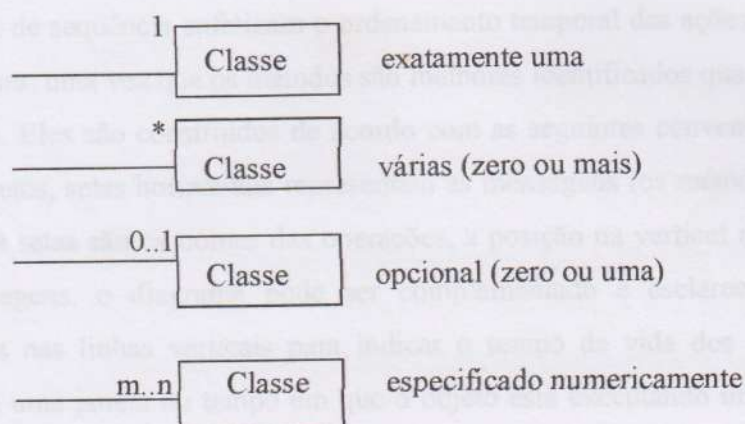


Figura 12. Classificação de Multiplicidades

Na figura 13, está um exemplo de um Diagrama de classe, para uma determinada venda.

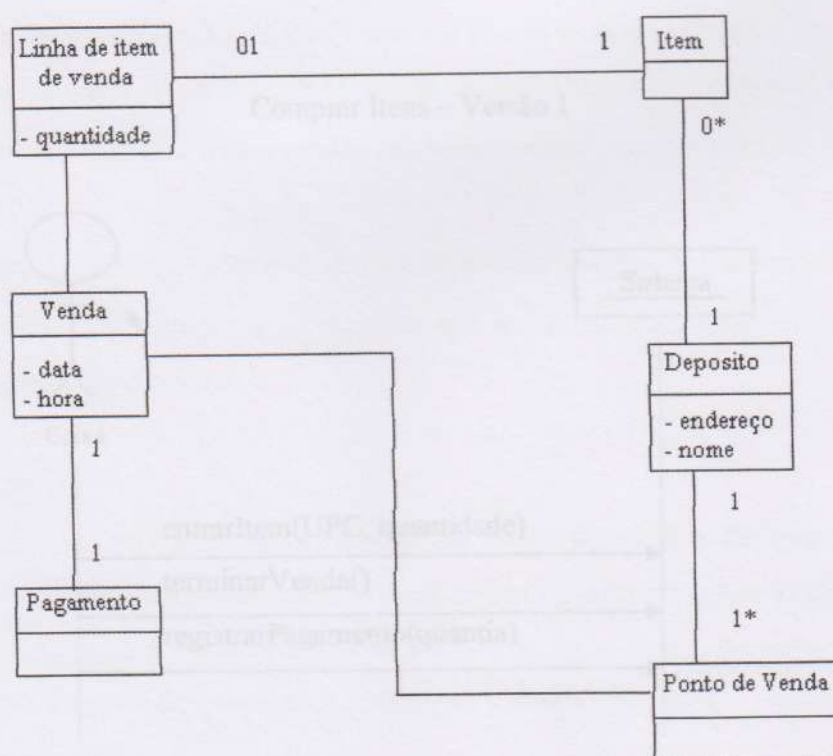


Figura 13: Exemplo de uma diagrama de classes.

2.3.3 Diagrama de Sequência

Os Diagramas de sequência enfatizam o ordenamento temporal das ações. Servem para validar o diagrama de classes, uma vez que os métodos são melhores identificados quando os objetos trocam mensagens entre si. Eles são construídos de acordo com as seguintes convenções: Linhas verticais representam os objetos, setas horizontais representam as mensagens (os métodos) passadas entre os objetos, rótulos das setas são os nomes das operações, a posição na vertical mostra o ordenamento relativo das mensagens, o diagrama pode ser complementado e esclarecido por anotações e retângulos situados nas linhas verticais para indicar o tempo de vida dos objetos (ativação). A ativação determina uma janela no tempo em que o objeto está executando uma ação, que pode ser seu próprio código ou aguardando o retorno de alguma mensagem enviada. Ocorre quando um objeto recebe uma mensagem. A linha da vida representa a existência do objeto em um dado momento. As mensagens são mostradas como setas (síncronas, assíncronas ou simples) entre as linhas da vida. A seguir um exemplo de um Diagrama de sequência, Versão 1 para uma Compra de Itens.

Comprar Itens – Versão 1

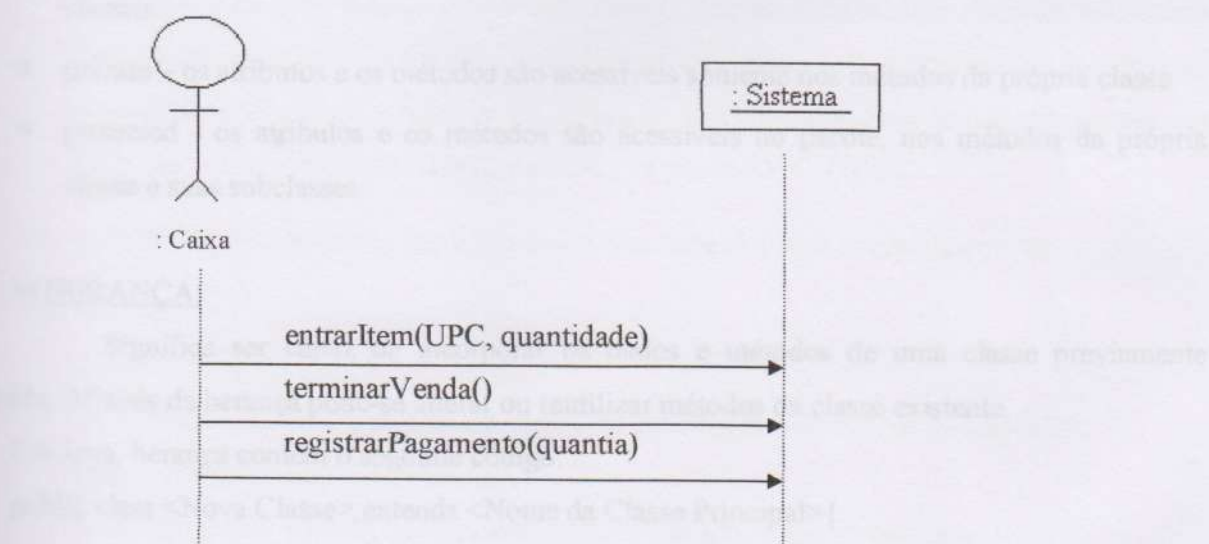


Figura 14. Exemplo do Diagrama de Sequência.

2.4 JAVA

A Linguagem Java surgiu em 1995.

Uma das vantagens da linguagem Java é a sua portabilidade, sua capacidade de ser deslocada entre sistemas diferentes.

Segundo Junior B. (2002) “Java é um programa elaborado em uma plataforma X (um sistema operacional qualquer) roda com a mesma eficiência em outra plataforma. Essa capacidade é conferida a um programa Java pela máquina virtual Java (Java Virtual Machine - JVM)”.

Java possibilita a implementação dos conceitos de orientação a objetos, tais como:

a) ENCAPSULAMENTO:

Propriedade de autonomia para definir o que o mundo externo ao objeto poderá visualizar e acessar. Implementado através de modificadores de acesso: os atributos de uma classe devem ser privados, e devem existir métodos get, ocorre quando se quer obter algum valor, e método set, ocorre quando se atribui valores, público para o acesso a esses atributos.

No encapsulamento, existem alguns tipos de modificadores de visibilidade, são eles:

- public - os atributos e os métodos são sempre acessíveis em todos os métodos de todas as classes.
- private – os atributos e os métodos são acessíveis somente nos métodos da própria classe.
- protected - os atributos e os métodos são acessíveis no pacote, nos métodos da própria classe e suas subclasses.

b) HERANÇA:

Significa ser capaz de incorporar os dados e métodos de uma classe previamente definida. Através da herança pode-se alterar ou reutilizar métodos da classe existente.

Em Java, herança contém o seguinte código:

```
public class <Nova Classe> extends <Nome da Classe Principal> {  
    ...  
}
```

c) CLASSE:

Um protótipo que define os atributos e métodos a todos os objetos de um determinado tipo, para servir como molde para a criação de objetos.

Em Java, uma classe contém o seguinte código:

```
public class <Nome da Classe> {  
    ...  
}
```

d) OBJETOS:

Conjuntos distintos de atributos que representam entidades do mundo real, implementados por software e gerados através de classes.

Exemplo: A classe Empregados, seus objetos são Maria, João que são empregados de uma determinada empresa.

Figura 13: Diagrama 3. JDBC: Ponte entre o Java e o Banco de dados

2.4.1 JDBC

A Microsoft possui uma interface chamada ODBC que dá suporte para sistemas acessarem Banco de dados relacionais como o Access e SQL Server. Java não interage diretamente com drivers padrões da plataforma Windows, a maneira mais fácil de um programa Java se comunicar com um Banco de dados que utiliza interface ODBC, é através da ponte JDBC-ODBC.

Para Criar uma conexão com o banco de dados são necessários os seguintes passos:

- Carregar o driver
- Criar a conexão:
- é necessário utilizar o método getConnection do DriverManager, especificando o nome da fonte de dados, um nome de usuário e senha.
- O nome da Fonte de dados ODBC possui a seguinte forma jdbc:odbc: nomeFonteODBC

A seguinte figura ilustra a arquitetura JDBC, delimitando o escopo da aplicação e da infraestrutura de acesso aos bancos de dados.

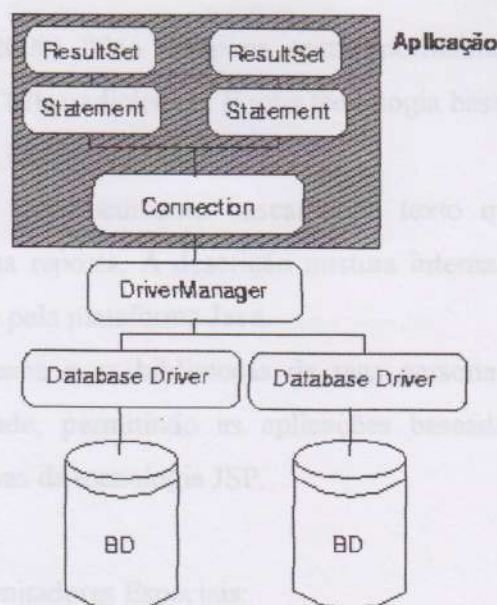


Figura 15. Estrutura do JDBC. Fonte tirada do site.

O JDBC estabelece uma conexão com o Banco de dados, envia requisições SQL e processa os resultados.

- **ResultSet** – É o objeto que o Java retorna de um comando SELECT.

Uma instância do resultset contém um conjunto de registros. Para acessar esses registros é necessário em cada linha retirar o seu valor de acordo com o seu tipo.

- O método **next** do ResultSet move o curso de registro em registro, O método **get** serve para recuperar o valor de cada coluna de acordo com o seu tipo.

- Um Statement Executa uma instrução SQL fixa e retorna um resultado. Por padrão, somente um resultado pode permanecer aberto por consulta.

2.4.2 JSP (Java Server Pages)

Segundo Fields (2000) “São arquivos texto, normalmente com a extensão “.jsp”, que substituem as paginas HTML tradicionais. É uma tecnologia baseada na linguagem Java, que facilita a construção de web sites dinâmicos.”

Uma página jsp é um documento baseado em texto que descreve como processar uma solicitação para criar uma reposta, A descrição mistura internamente dados modelares com ações dinâmicas e é alavancada pela plataforma Java.

Há ainda o JavaBeans e as bibliotecas de tags personalizadas que são componentes que encapsulam funcionalidade, permitindo as aplicações baseadas em componentes. Javabeans e bibliotecas são partes ativas da tecnologia JSP.

- Existem os delimitadores Especiais:

Os tags Delimitadores para a inclusão de scriptlets são: `<%...%>`.

`<%! -` para a declaração de atributos e de métodos.

Exemplo: `<% ! public String nome; %>`

`<%@ -` para utilização de diretivas, como por exemplo, as diretivas page, include e taglib.

Exemplo: `<% @ include file= “nome do Arquivo” %>`

`<%= -` para imprimir na tela o resultado da expressão.

Exemplo: `<% = Fatorial (x) %>`

Objetos Implícitos:

São objetos disponíveis para o desenvolvedor da JSP, relativos a cada uma JSP criada.

- request – dados de Solicitação
- response – dados de Resposta
- out – fluxo de Saída
- session – dados de Sessão específicos de usuário.
- exception – erros não capturados ou exceção.

Os objetos implícitos podem ser divididos em quatro categorias principais:

- Relacionados ao servlet
- Ligados à entrada e saída.
- Informações de contexto
- Objetos resultantes de erro.

Exemplo de uma Jsp, para alterar dados de uma agenda:

```
<!-- AlterarDados.jsp -->
<HTML>
  <HEAD><TITLE> Formulário de Alteração de dados </TITLE></HEAD>
  <BODY>
    <%String nome = request.getParameter("nome");
    Agenda ag=new Agenda();
    Contato Atual=ag.recuperaContato(nome);
    if (atual != null){ %>
      <FORM ACTION="altera.jsp" METHOD="post">
        Nome:<INPUT TYPE="text" NAME="Cnome" VALUE="<%=atual.getNome()%>">
        <BR>
        End:<INPUT TYPE="text" NAME="Cend" VALUE="<%=atual.getEndereco()%>">
        <BR>
        <INPUT TYPE="hidden" NAME="original" VALUE="<%=nome%>">
        </FORM>
        <%} else { %>
          Não existe contato com tal nome !
        <% } %>
      </BODY>
    </HTML>
```

Figura 16. Exemplo de uma Jsp.

2.4.3 Servlets

São componentes, pequenos programas na linguagem Java, que geram as páginas visualizadas pelo navegador. Sua grande vantagem está na velocidade após a primeira execução. Podem funcionar também como componentes de aplicação, dando suporte a outros servlets ou a páginas JSP.

São classes Java e como tal desfrutam todos os mecanismos e ferramentas da plataforma Java.

2.4.4 Javabeans

São componentes de uma aplicação, sua função é encapsular funções, separando a lógica de programação da apresentação. Um conjunto de tags predefinidas permite ao programador visual da página acessar os recursos disponibilizados pelos Beans, sem que muitas vezes sejam necessárias a escrita de uma única linha de código Java.

Para uma JSP, um Bean é simplesmente uma classe Java que pode ser acessada por tags específicos de modo a receber dados de uma página ou enviá-los a uma página.

Tipos diferentes de JavaBeans:

- Bean de componente visual

Recursos úteis para criar a interface com o usuário.

- Bean de dados

Maneira para acessar dados.

- Beans de serviços

Fornecer acesso a um serviço particular, exemplo: acessar um banco de dados.

- Beans de JSP

Três tags simples permitem a interação com JavaBeans:

- Colocar Beans na página.

```
<jsp:useBean id="Nome do Bean" class="nome da classe"/>
```

- Acessar suas propriedades: <jsp:getProperty... >

- Modificar suas propriedades: <jsp:setProperty...>

2.5 BANCO DE DADOS

2.5.1 Sql (Structured Query Language)

Segundo Rocha (2003), "Linguagem de Consulta Estruturada Implementa as operações de Álgebra Relacional".

É Subdividida em três partes:

DDL (DATA DEFINITION LANGUAGE): Comandos de definições de dados.

DML (DATA MANIPULATION LANGUAGE): Comandos de manipulação de dados.

DPL (DATA PROGRAMING LANGUAGE): Comandos para programação estruturada: Laços, etc.

SINTAXE DOS CAMANDOS SQL:

Listar os atributos desejados no resultado de uma consulta:

SELECT [campos que deverão ser exibidos]

FROM [Tabelas que tem as informações]

WHERE [Condição da pesquisa, mais a condição de amarração de tabelas]

Inserir dados na tabela:

INSERT INTO nome da tabela (atributos)

VALUES (valores para estes atributos)

Remover dados de uma tabela:

DELETE nome da tabela que será removido

FROM tabela

WHERE atributo da tabela

Criar tabelas:

CREATE TABLE nome da tabela

(atributo e tipo).

PRIMARY KEY (atributo) REFERENCES (tabela),

FOREIGN KEY (atributo) REFERENCES (tabela);

3. ANÁLISE DO SISTEMA DE ACOMPANHAMENTO DE PROJETOS VIA WEB

3.1 DOCUMENTO DE REQUISITOS

Objetivos Gerais

A finalidade deste sistema é desenvolver uma solução para o acompanhamento de projetos para ser utilizado em instituições de ensino e pesquisa. Podendo ser utilizados, por exemplo, para o acompanhamento de projetos de final de curso dos Alunos da FARN.

Cliente

Faculdade Natalense para o Desenvolvimento do Rio Grande do Norte.

Objetivos Específicos

Em Geral, o objetivo é facilitar o relacionamento de um Orientador com seus orientandos, na execução de um determinado projeto. Mais especificamente, estes objetivos incluem:

- Controle das tarefas de um orientando

Funções do Sistema

Ref #	Função	Categoria
R1.1	Listar tarefas em atraso	evidente
R1.2	Registrar e finalização de tarefas	evidente
R1.3	Exibir a descrição de uma tarefa para o Orientando.	evidente
R1.4	Exibir descrição de uma tarefa para ser Alterada pelo orientador	evidente
R1.5	Registrar tarefas atribuídas	evidente
R1.6	Listar dúvidas de um orientando	evidente

R1.7	Registrar a resposta a uma dúvida do orientando	evidente
R1.8	Cadastrar Projeto	evidente
R1.9	Cadastrar Orientando	evidente
R1.10	Cadastrar Orientador	evidente
R1.11	Alterar e remover Projeto	evidente
R1.12	Alterar e remover Orientando	evidente
R1.13	Alterar e remover Orientador	evidente

Figura 17. Funções do sistema.

Atributos do sistema

Atributo	Detalhes e Restrições de Limites
Metáfora de Interface	gráfica, colorida
Plataformas de Sistemas	Internet Explorer
Operacionais - Servidor	contêiner Jsp/Servlet + (Resin, Tomcat, etc)
- Cliente	Navegador Internet

Figura 18. Atributos do sistema.

3.2 DIAGRAMA DE CASOS DE USO

Para modelar a funcionalidade do sistema utilizamos um Diagrama de Casos de uso. Na figura 19, está o Diagrama em questão.

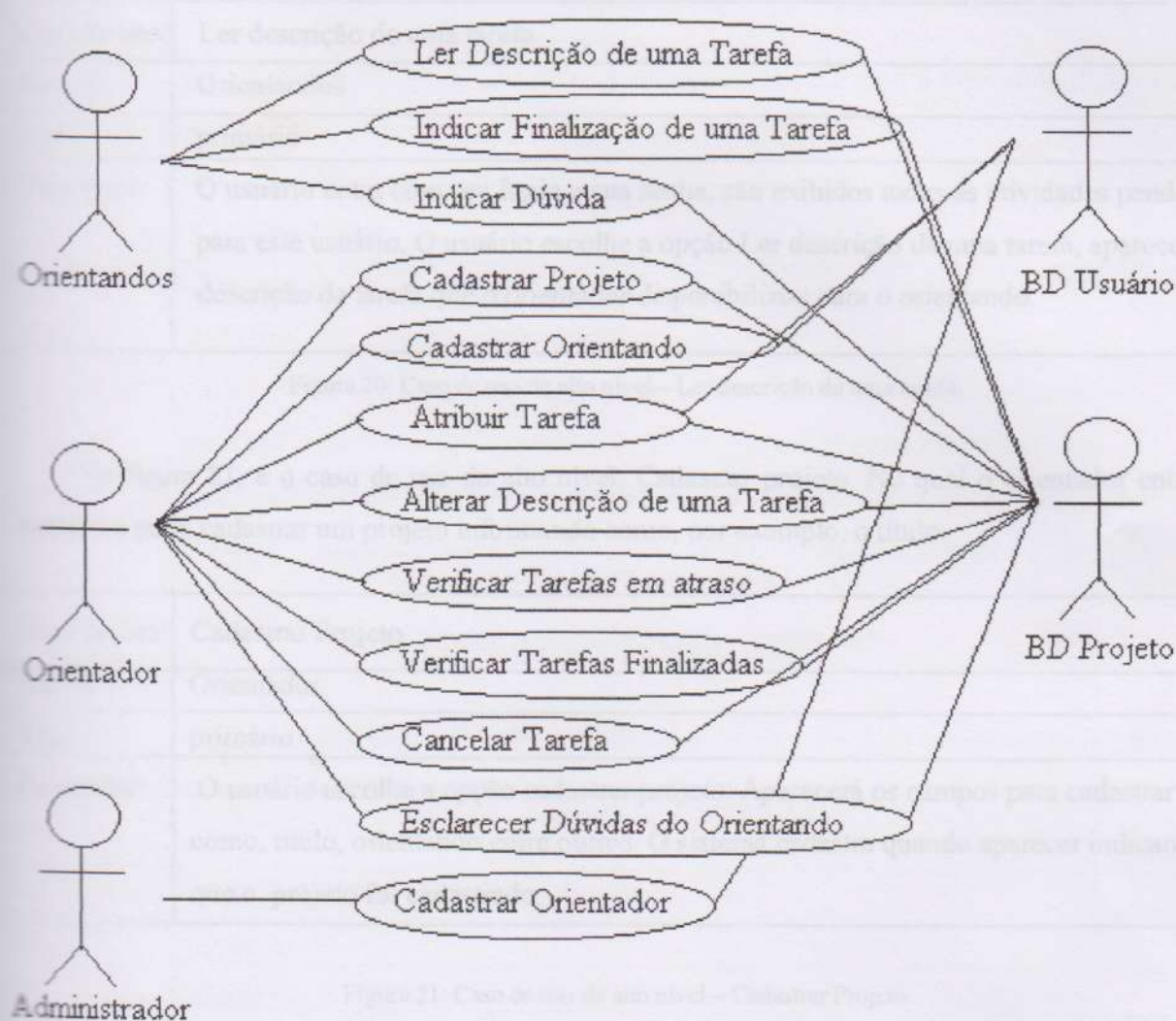


Figura: 19 Diagrama de casos de uso do sistema.

3.3 CASOS DE USO DE ALTO NÍVEL

Para modelar a funcionalidade de cada caso de uso, utilizamos casos de uso de alto nível.

Na figura 20, é o caso de uso de alto nível: Ler a descrição de uma tarefa. No qual o orientando entra no sistema e pode escolher a opção para ler a descrição de uma tarefa.

Figura 20: Caso de uso de alto nível – Ler descrição de uma tarefa.

Caso de uso:	Ler descrição de uma tarefa
Atores:	Orientandos
Tipo:	primário
Descrição:	O usuário entra com seu login e sua senha, são exibidos todas as atividades pendentes para este usuário. O usuário escolhe a opção Ler descrição de uma tarefa, aparecerá a descrição da tarefa que o orientador disponibilizou para o orientando.

Figura 20: Caso de uso de alto nível – Ler descrição de uma tarefa.

Na figura 21, é o caso de uso de alto nível: Cadastrar projeto. No qual o orientador entra no sistema e pode cadastrar um projeto informando como, por exemplo, o título.

Caso de uso:	Cadastrar Projeto
Atores:	Orientador
Tipo:	primário
Descrição:	O usuário escolhe a opção cadastrar projeto. Aparecerá os campos para cadastrar como, título, orientando entre outros. O sistema cadastra quando aparecer indicando que o projeto foi cadastrado.

Figura 21: Caso de uso de alto nível – Cadastrar Projeto.

Na figura 22, é o caso de uso de alto nível: Indicar finalização de uma tarefa. No qual o orientando indica ao Orientador quando terminar uma determinada tarefa.

Caso de uso:	Indicar finalização de uma tarefa
Ator:	Orientandos

Tipo:	primário
Descrição:	O orientando entra com seu login e sua senha, são exibidos todas as atividades pendentes para este usuário. O orientando ao terminar uma tarefa, deve indicar para o orientador utilizando a opção indicar finalização de uma tarefa.

Figura 22: Caso de uso de alto nível – Indicar finalização de uma tarefa.

Na figura 23, é o caso de uso de alto nível: Cancelar tarefa. No qual o orientador pode cancelar uma determinada tarefa.

Caso de uso:	Cancelar Tarefa
Ator:	Orientador
Tipo:	primário
Descrição:	O orientador entra com seu login e sua senha, é exibida uma lista de projetos orientados por este. Após selecionar um projeto em questão são exibidos os orientandos e suas tarefas. Haverá uma opção cancelar tarefa, onde tendo uma tarefa ao qual o orientador não precisará passar para o orientando, poderá ser apagada.

Figura 23: Caso de uso de alto nível – Cancelar tarefa.

Na figura 24, é o caso de uso de alto nível: Indicar dúvida. No qual o orientando informará ao Orientador quando estiver com dúvida em alguma tarefa.

Caso de uso:	Indicar dúvida
Ator:	Orientando
Tipo:	primário
Descrição:	O orientando entra com seu login e sua senha, são exibidos todas as atividades pendentes para este usuário, Este ao ter alguma duvida em relação a tarefa deverá escolher a opção indicar dúvida, que este informará ao orientador

Figura 24: Caso de uso de alto nível – Indicar dúvida.

Na figura 25, é o caso de uso de alto nível: Alterar descrição de uma tarefa. No qual o orientador pode alterar uma descrição de uma tarefa.

Caso de uso:	Alterar descrição de uma tarefa
Ator:	Orientador
Tipo:	primário
Descrição:	O orientador entra com seu login e sua senha, é exibida uma lista de projetos orientados por este. Após selecionar um projeto em questão são exibidos os orientandos e suas tarefas. Ao entrar no sistema o orientador terá a opção Alterar descrição de uma tarefa, caso ele queira alterar.

Figura 25: Caso de uso de alto nível – Alterar descrição de uma tarefa.

Na figura 26, é o caso de uso de alto nível: Verificar tarefas em atraso. No qual é exibida uma lista de tarefas em atraso.

Caso de uso:	Verificar tarefas em atraso
Ator:	Orientador
Tipo:	primário
Descrição:	O orientador entra com seu login e sua senha, é exibida uma lista de projetos orientados por este. Após selecionar um projeto em questão são exibidos os orientandos e suas tarefas. Este escolhe a opção verificar tarefas em atraso, onde aparece a descrição da tarefa, o orientando e a data que tinha que ser entregue a tarefa.

Figura 26: Caso de uso de alto nível – Verificar tarefas em atraso.

Na figura 27, é o caso de uso de alto nível: Verificar tarefas finalizadas. No qual o orientando informará ao Orientador quando tiver finalizado uma tarefa.

Caso de uso:	Verificar tarefas finalizadas
Ator:	Orientador
Tipo:	primário
Descrição:	O orientador entra com seu login e sua senha, é exibida uma lista de projetos orientados por esta. Após selecionar um projeto em questão são exibidos os orientandos e suas tarefas. Escolhendo a opção tarefas finalizadas, aparecerá as tarefas que o orientando finalizou.

Figura 27: Caso de uso de alto nível – Verificar tarefas finalizadas.

Na figura 28, é o caso de uso de alto nível: Atribuir tarefa. No qual o orientador atribuirá uma tarefa ao Orientando.

Caso de uso:	Atribuir Tarefa
Ator:	Orientador
Tipo:	primário
Descrição:	O orientador entra com seu login e sua senha, é exibida uma lista de projetos orientados por esta. Após selecionar um projeto em questão são exibidos os orientandos e suas tarefas. Ao escolher a opção atribuir tarefa, o orientador atribuirá uma determinada tarefa a um orientando.

Figura 28: Caso de uso de alto nível – Atribuir tarefa.

3.5 CASOS DE USO EXPANDIDO E DIAGRAMA DE SEQUÊNCIA

Caso de Uso Expandido

Para modelar a funcionalidade de cada caso de uso, a função de cada ator e as ações do sistema utilizamos casos de uso Expandido.

Na figura 29, é o Caso de uso expandido: Cadastrar Projeto. No qual o orientador poderá escolher a opção para cadastrar um determinado projeto.

Caso de uso: Cadastrar Projeto Atores: Orientador Finalidade: Cadastrar um determinado projeto no sistema Visão Geral: O usuário entra no sistema, escolhe a opção cadastrar projeto, onde aparecerá os campos para preenchido como, título, data de início e data final. O sistema cadastra quando aparecer uma mensagem informando que foi efetuado o cadastramento. Tipo: primário Referências Cruzadas: R1.8	
Ação do Ator	Resposta do sistema
1. Este caso de uso começa quando o orientador entra no sistema e escolhe a opção cadastrar projeto.	2. Determina se o orientador é cadastrado ou não.
3. O sistema registra cada Projeto	4. Indica se o projeto já está cadastrado ou não.
	5. Informa o cadastramento do projeto.

Figura 29: Caso de uso expandido – cadastrar projeto.

Na figura 30, é o Caso de uso expandido: Indicar finalização de uma tarefa. No qual o orientando indicará para o Orientador quando tiver finalizado uma determinada tarefa.

Caso de uso: Indicar finalização de uma tarefa

Atores: Orientandos

Finalidade: Indicar quando uma tarefa for finalizada para o orientador.

Visão Geral: O orientando entra no sistema com seu login e sua senha, são exibidos todas as atividades pendentes para este usuário. O orientando ao terminar uma tarefa, deve indicar para o Orientador utilizando a opção indicar finalização de uma tarefa.

Tipo: primário

Referências Cruzadas: R1.2

Ação do Ator

Resposta do sistema

- | | |
|---|---|
| 1. Este caso de uso começa quando o orientando entra no sistema e escolhe a opção Indicar Tarefa finalizada, quando este concluir uma Tarefa. | 2. Determina se o orientando é cadastrado ou não. |
| 3. O orientando indica quando finalizar uma tarefa. | 4. Indica se o projeto já está finalizado ou não |
| | 5. Informa a finalização da tarefa. |

Na figura 32, Figura 30: Caso de uso Expandido - Indicar finalização de uma tarefa.

Na figura 31, é o Caso de uso expandido: Indicar dúvida. No qual o orientando indicará para o Orientador quando tiver dúvida em uma determinada tarefa.

Caso de uso: Indicar dúvida

Atores: Orientando

Finalidade: Indicar ao orientador quando tiver uma dúvida na tarefa.

Visão Geral: O orientando entra no sistema com seu login e sua senha, são exibidos todas as atividades pendentes para este usuário. Caso este tenha alguma dúvida em relação à tarefa, então deverá escolher a opção indicar dúvida, no qual o sistema informará ao orientador.

Tipo: primário

Referências Cruzadas: R1.6

Ação do Ator	Resposta do sistema
2. Este caso de uso começa quando o orientando entra no sistema e escolhe a opção Indicar Dúvida.	2. Determina se o orientando é cadastrado ou não.
3. O orientando indica quando estiver alguma dúvida.	4. Informa a dúvida ao Orientador.

Figura 31: Caso de uso Expandido – Indicar dúvida.

• Diagrama de Seqüência

Para modelar a funcionalidade de cada caso de uso expandido, utilizamos o Diagrama de seqüência.

Indicar Finalização de uma Tarefa

Na figura 32, é o diagrama de seqüência, fase 1, para o caso de uso Indicar tarefa finalizadas.

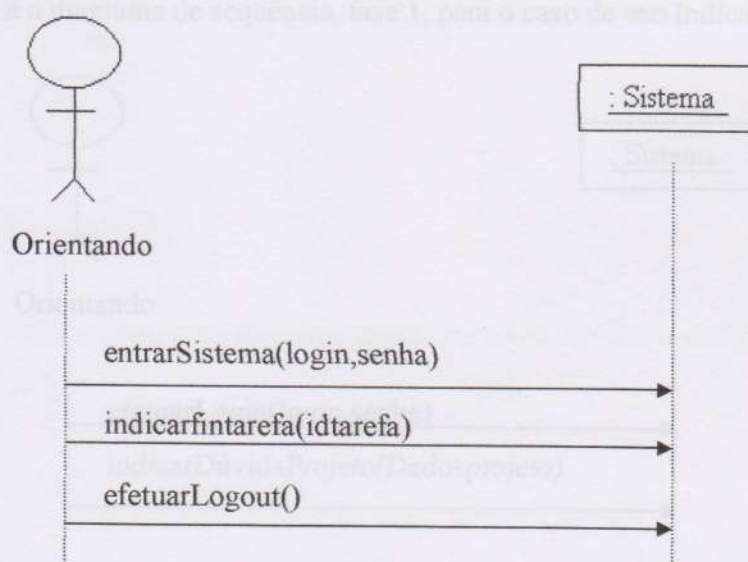


Figura 32: Diagrama de seqüência – Indicar finalização de uma tarefa.

Cadastrar Projeto

Na figura 33, é o diagrama de seqüência, fase 1, para o caso de uso Cadastrar Projeto.

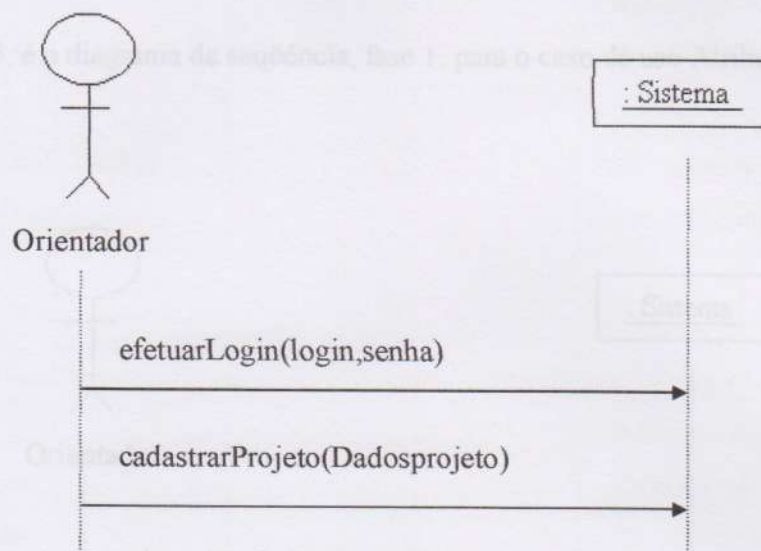


Figura 33: Diagrama de seqüência – Cadastrar Projeto.

Indicar Dúvida

Na figura 34, é o diagrama de seqüência, fase 1, para o caso de uso Indicar dúvida.

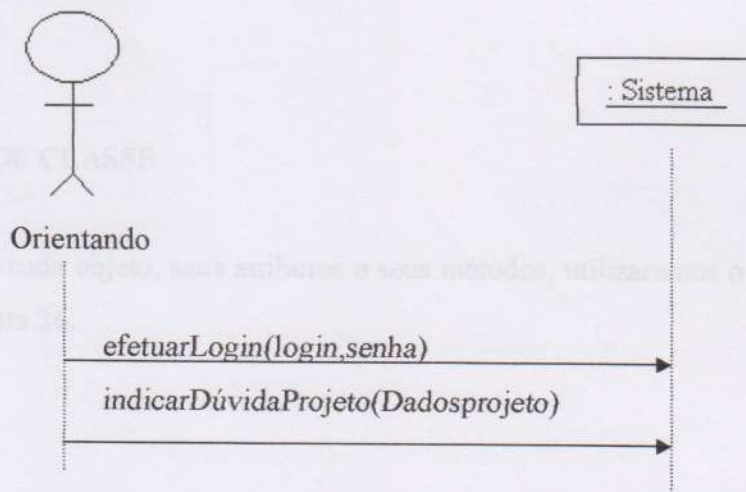


Figura 34: Diagrama de seqüência – Indicar dúvida.

Atribuir Tarefa

Na figura 35, é o diagrama de sequência, fase 1, para o caso de uso Atribuir tarefa.

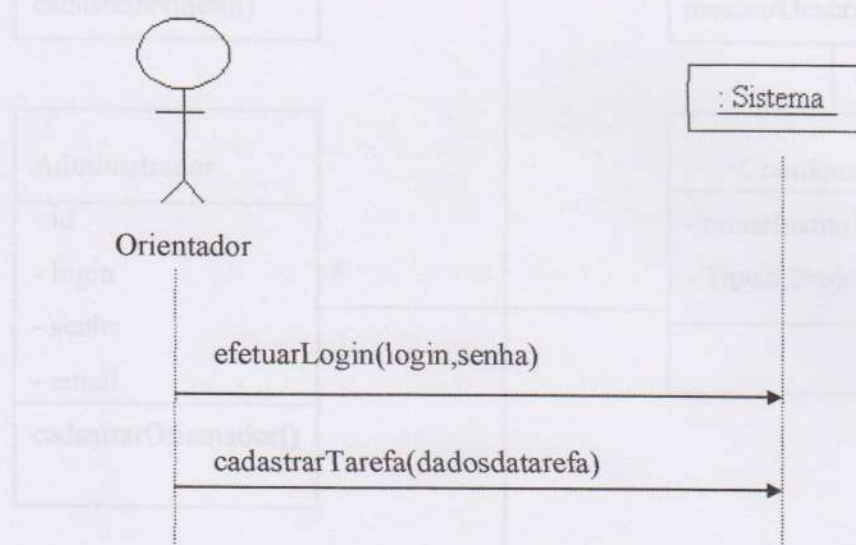


Figura 35: Diagrama de sequência – Atribuir tarefa.

3.6 DIAGRAMA DE CLASSE

Para modelar cada objeto, seus atributos e seus métodos, utilizaremos o Diagrama de Classes, como mostra a figura 36.

Figura 36: Diagrama de Classes do Sistema

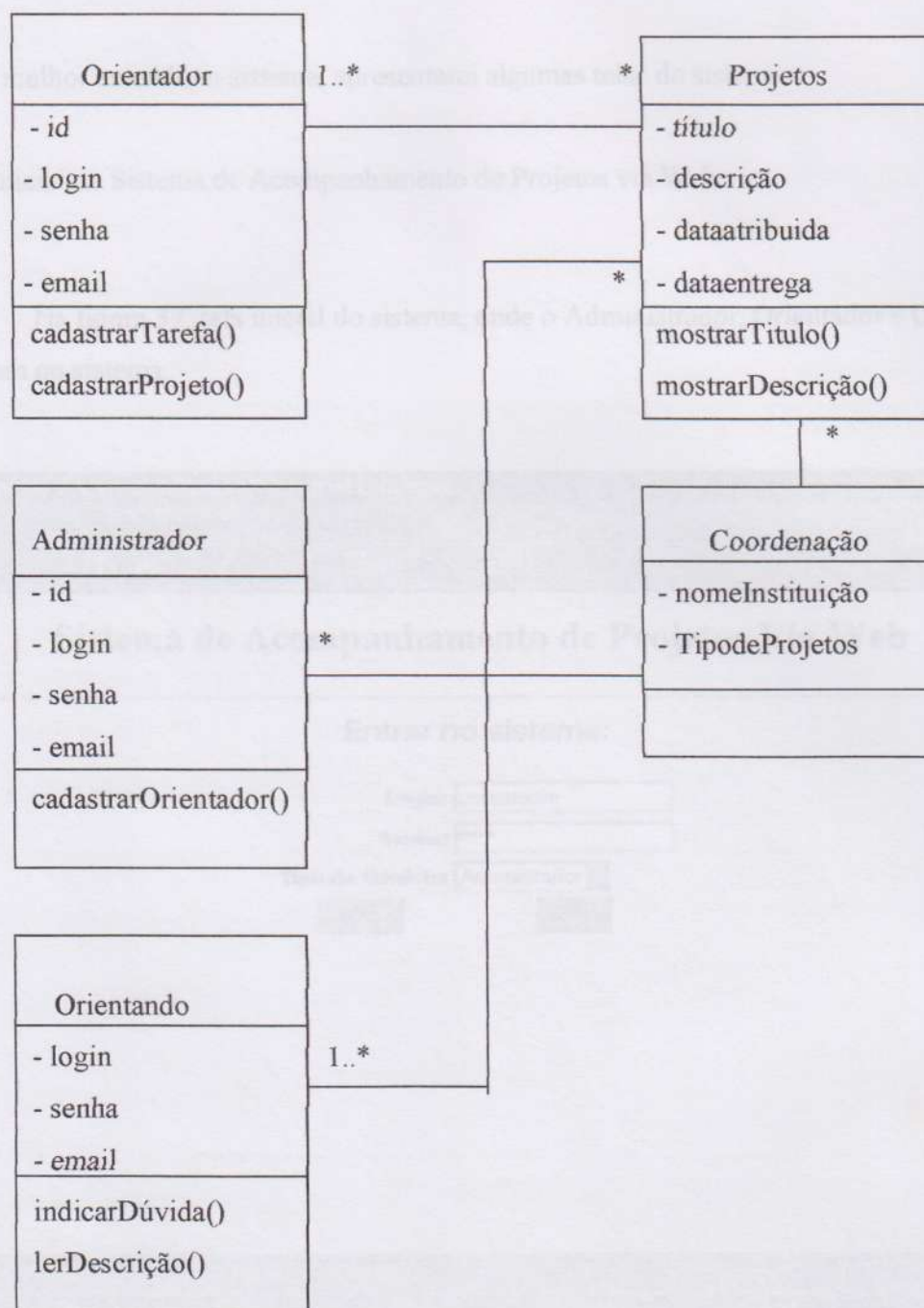


Figura 36- Diagrama de Classes do Sistema.

4. IMPLEMENTAÇÃO DO SISTEMA DE ACOMPANHAMENTO DE PROJETOS VIA WEB

Para melhor entender o sistema, apresentarei algumas telas do sistema.

4.1 Tela inicial do Sistema de Acompanhamento de Projetos via Web.

Na figura 37, tela inicial do sistema, onde o Administrador, Orientador e Orientandos entram no sistema.

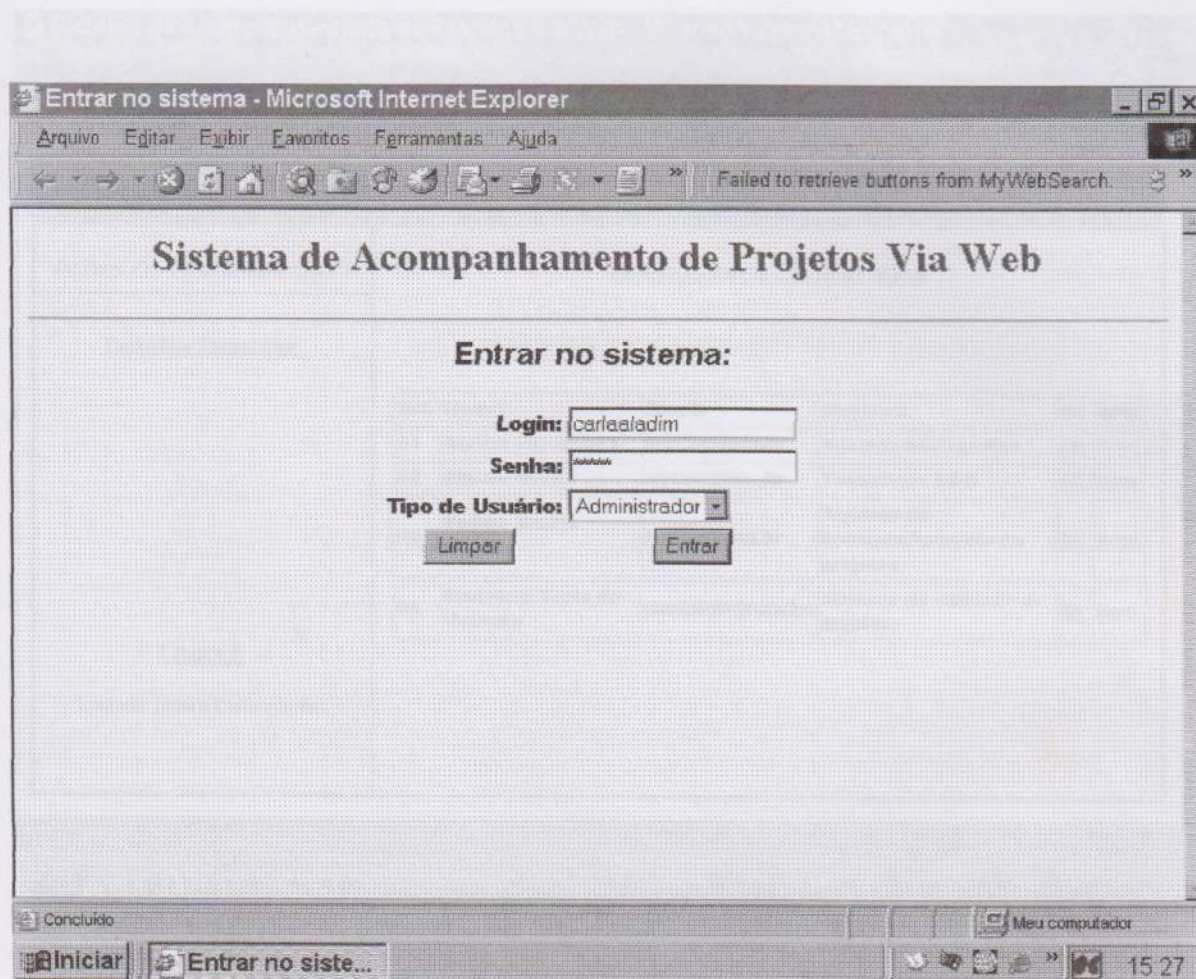


Figura 37: Tela inicial do sistema.

4.2 Tela do Administrador

Na figura 38, tela do Administrador do sistema, onde o Administrador entra no sistema e verifica suas funções.

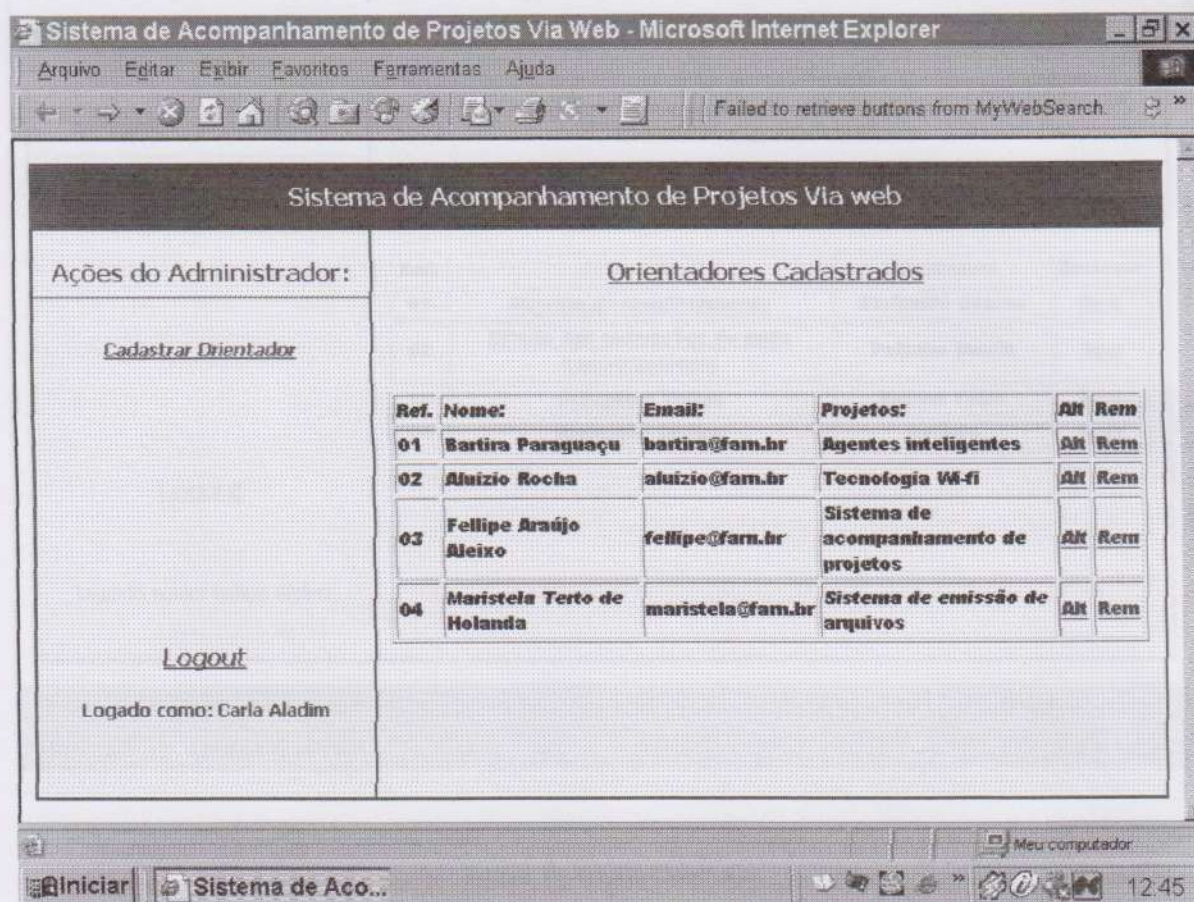


Figura 38: Tela do Administrador.

4.4 Tela para cadastrar os projetos

Na figura 40, é a tela para cadastrar projetos no sistema, o orientador entra no sistema e escolhe a opção desejada.

The screenshot shows a web browser window titled "Cadastrar Projeto - Microsoft Internet Explorer". The address bar shows a failed attempt to retrieve buttons from MyWebSearch. The main content area is titled "Sistema de Acompanhamento de Projetos Via web" and is divided into two columns. The left column, titled "Ações do Orientador:", contains links for "Cadastrar Projeto", "Logout", and "Logado como: Fellipe Aleixo". The right column, titled "Cadastrar Projeto", contains a form with three input fields: "Titulo:" (containing "sistema de acompanhamento de projetos"), "Inicio:" (containing "01/01/2004"), and "Fim:" (containing "01/12/2004"). Below these fields are two buttons: "Limpar" and "Cadastrar". The status bar at the bottom shows "Concluído" and "Meu computador". The taskbar at the very bottom shows the "Iniciar" button and the "Cadastrar Proje..." application icon, along with a clock showing 16:00.

Sistema de Acompanhamento de Projetos Via web	
Ações do Orientador:	Cadastrar Projeto
Cadastrar Projeto	Título: sistema de acompanhamento de projetos
	Início: 01/01/2004
	Fim: 01/12/2004
	Limpar Cadastrar
Logout	
Logado como: Fellipe Aleixo	

Figura 40: tela para cadastrar projeto no sistema.

4.5 Tela para cadastrar os projetos

Na figura 41, é a tela para o Orientando indicar uma tarefa quando esta for concluída.

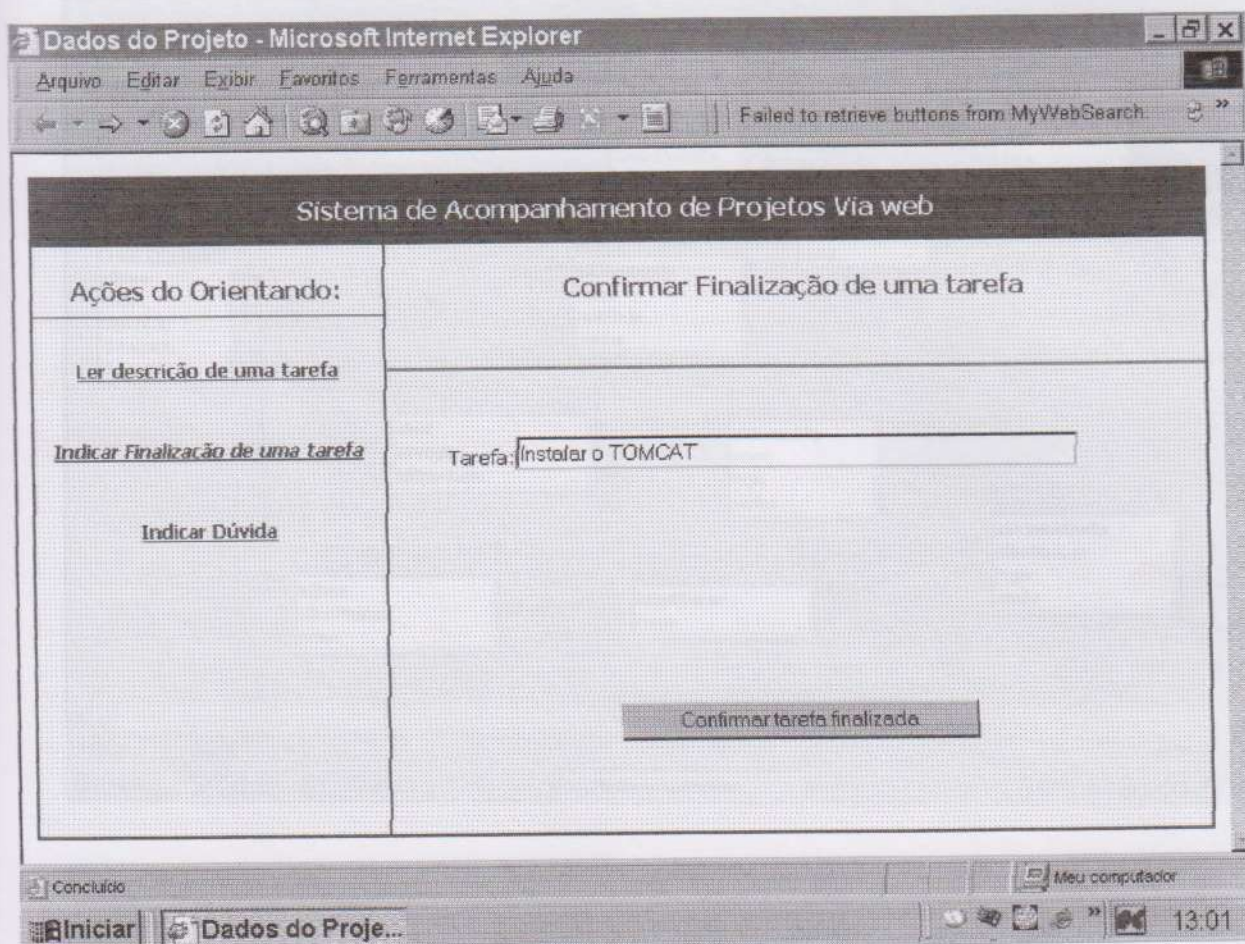


Figura 41: tela para confirmar tarefa finalizada.

4.5 Tela do Banco de Dados do Sistema de Acompanhamento de Projetos Via Web no Access.

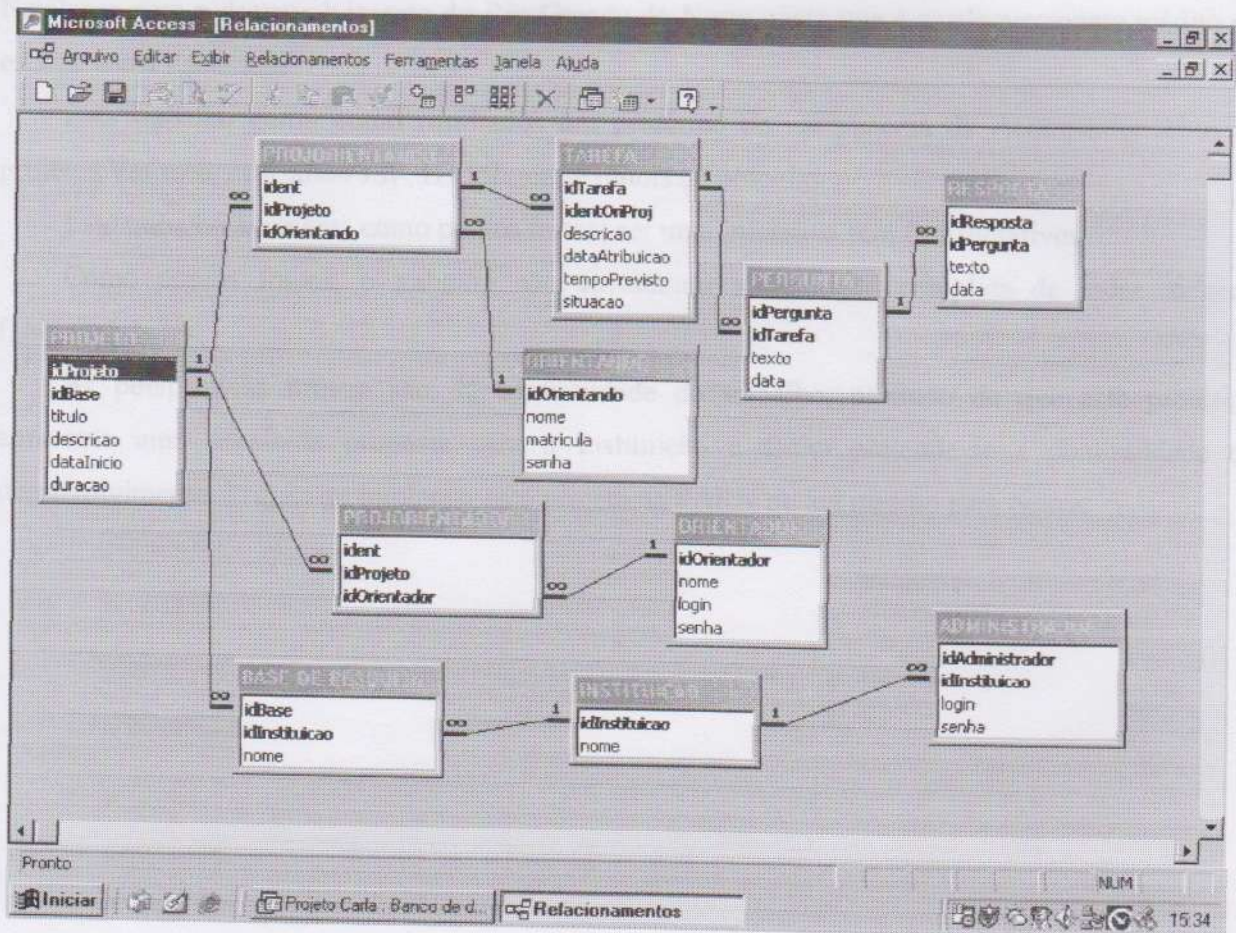


Figura 42: tela do Banco de dados.

5. CONCLUSÃO

Ao término desta etapa do trabalho, conclui-se que este é perfeitamente viável para a Faculdade Natalense para o desenvolvimento do Rio Grande do Norte, visto que é academicamente inédito no espaço desta Faculdade.

Este trabalho gerou como resultados um protótipo de um sistema de Acompanhamento de projetos Via web, utilizando JSP, HTML, entre outras tecnologias.

Este trabalho apresenta como pontos fortes; ser uma iniciativa sem fins lucrativos.

Como pontos fracos, o trabalho não apresentou a descrição completa de todas as suas funcionalidades.

As perspectivas futuras são de continuidade do trabalho, partindo de que este protótipo apresenta uma excelente proposta para a Instituição e assim pretende-se a continuidade do desenvolvimento. Projeta-se implantar este projeto na FARN ou até mesmo para outros.

REFERÊNCIAS BIBLIOGRÁFICAS

ALCÂNTARA, Andréia Almeida de. et. al. **Home Pages** - Recursos e Técnicas para criação de páginas na www. 2ª Edição, Rio de Janeiro: Campus, 1997.

COMER, Douglas E. **Redes de computadores e Internet**. 2ª Edição. Editora Bookman, 1999.

FIELDS, Duane K; KOLB, Mark A. **Desenvolvendo na Web com Java Server Pages**, Trad. Rejane Freitas. Rio de Janeiro: Editora Ciência Moderna, 2000.

FOWLER, Martin; SCOTT, Kendall. **UML Essencial** – Um breve guia para a linguagem padrão de modelagem de Objetos. 2ª Edição, Porto Alegre: Bookman, 2000.

JUNIOR, Francisco B. **JSP – Java Server Pages** – A Tecnologia Java na Internet, São Paulo: Editora Érica, 2002.

MACORATTI, José Carlos. **Modelando Sistemas em UML** - Casos de Uso. < www.macoratti.net/net_uml2.htm > Acesso em : 10 de Novembro de 2004.

MILLER, Michael. **Internet**, rápido e fácil para iniciantes, Rio de Janeiro: Editora Campus, 2ª Edição, 1998.

RICARTE, Ivan Luiz Marques. **Interface entre Java e Banco de dados**, São Paulo: UNICAMP, < <http://www.dca.fee.unicamp.br/cursos/PooJava/javadb/jdbc.html> >. Acesso em: 23 maio 2004. Última atualização: 15 De outubro 2002.

ROCHA, Aluizio. **Apostila de Redes de Computadores**, Natal: FARN, 2º Semestre, 2002.

ROCHA, Aluizio. **Apostila de Banco de Dados II**, Natal: FARN, 2º Semestre, 2003.

RANDALL, Neil. **Aprenda em 1 semana Internet**, Rio de Janeiro: Editora Campus, 1998.

BIBLIOGRAFIA CONSULTADA

ALCÂNTARA, Andréia Almeida de. et. al. **Home Pages** - Recursos e Técnicas para criação de páginas na www. 2ª Edição, Rio de Janeiro: Campus, 1997.

COMER, Douglas E. **Redes de computadores e Internet**. 2ª Edição. Editora Bookman, 1999.

FIELDS, Duane K; KOLB, Mark A. **Desenvolvendo na Web com Java Server Pages**, Trad. Rejane Freitas. Rio de Janeiro: Editora Ciência Moderna, 2000.

FOWLER, Martin; SCOTT, Kendall. **UML Essencial** – Um breve guia para a Linguagem padrão de modelagem de Objetos. 2ª Edição, Porto Alegre: Bookman, 2000.

HEIDE, Ann; STILBORNE, Linda. **Guia do Professor para a Internet**, Porto Alegre: Editora Artmed, 2ª Edição, 2000.

HONEYCUTT, Jerry. **Usando a Internet**, Rio de Janeiro: Editora Campus, 1998.

JUNIOR, Franscisco B. **JSP – Java Server Pages** – A Tecnologia Java na Internet, São Paulo: Editora Érica, 2002.

KORTH, Henry F; SILBERSCHATZ, Abraham. **Sistema de Banco de Dados**, São Paulo: Makron Books, 2ª Edição, 1995.

MACORATTI, José Carlos. Modelando Sistemas em UML - **Casos de Uso**. < www.macoratti.net/net_uml2.htm > Acesso em : 10 de Novembro de 2004.

MILLER, Michael. **Internet**, rápido e fácil para iniciantes, Rio de Janeiro: Editora Campus, 2ª Edição, 1998.

RANDALL, Neil. **Aprenda em 1 semana Internet**, Rio de Janeiro: Editora Campus, 1997.

RICARTE, Ivan Luiz Marques. **Interface entre Java e Banco de dados**, São Paulo: UNICAMP, <
<http://www.dca.fee.unicamp.br/cursos/PooJava/javadb/jdbc.html> >. Acesso em: 23 maio 2004.
 Última atualização: 15 De outubro 2002.

ROCHA, Aluizio. **Apostila de Redes de Computadores**, Natal: FARN, 2º Semestre, 2002.

ROCHA, Aluizio. **Apostila de Banco de Dados II**, Natal: FARN, 2º Semestre, 2003.

TORRES, Gabriel, **Redes Locais: Topologias e Periféricos**, Rio de Janeiro: Clube do Hardware, <
<http://www.clubedohardware.com.br/aula5.html> >. Acesso em: 26 junho 2004. No ar desde 01 de
 junho de 1998.

ANEXO I

Códigos de Jsp

Autentica.jsp

```

<%@ page import="br.farn.carla.*" %>
<%
String tipo = request.getParameter("tipo");
String login = request.getParameter("login");
String senha = request.getParameter("senha");

if ( login!=null && !login.equals("") && senha!=null && !senha.equals("") ) {
    ControleProjeto controle = new ControleProjeto();
    if ( tipo.equals("adm") ) {
        ViewAdministrador adm = controle.autenticaAdm(login, senha);
        if ( adm == null )
            response.sendRedirect("index.jsp?msg=err_autent");
        session.setAttribute("controleAdm",adm);
        session.setAttribute("login",login);
    }
} else {
    response.sendRedirect("index.jsp?msg=err_param");
}
%>
<html>
<frameset rows="2,11">
    <frame src="topo.html" name="Topo" scrolling="no">
</frameset>
<frameset cols="100,290">
    <frame src="menu.jsp" name="Menu" scrolling="no">
    <frame src="principal.jsp" name="Central">
</frameset>
</frameset>
</html>

```

Controle Sessão

```

<%
String login = (String) session.getAttribute("login");
if ( login == null ) {
    response.sendRedirect("index.jsp?msg=autenticar");
}
%>

```

Index.jsp

```

<HTML>
<HEAD></HEAD><TITLE>Entrar no sistema</TITLE>
<body bgcolor="#800000">

<B><font size="5" face="tahoma">
<p align="center"><font color="ffffff">
Sistema de Acompanhamento de Projetos Via Web</font>
</B></font></p>

<table width="300" bgcolor="beige" align="center">
  <form action="autentica.jsp" method="POST">
    <tr>
      <th colspan="2"><font size="4" face="arial">Entrar no sistema</font>
      <input type="hidden" name="tipo" value="adm">
    </tr>
    <tr>
      <td align="right">Login:
      <td><input type="text" name="login">
    </tr>
    <tr>
      <td align="right">Senha:
      <td><input type="password" name="senha">
    </tr>
    <tr>
      <td align="right">Tipo de Usuário:
      <td><select name="tipo">
        <option value="adm">Administrador
        <option value="prof">Orientador
        <option value="aluno">Orientando
      </select>
    </tr>
    <tr>
      <td align="center"><input type="reset" value="Limpar">

      <td colspan="2" align="center"><input type="submit" value="Entrar">
    </tr>
  </form>
</table>
</BODY>
</HTML>

```

Logout.jsp

```
<%
    session.invalidate();
    response.sendRedirect("index.jsp");
%>
```

Menu.jsp

```
<%@ include file="controleSessao.jsp" %>

<html>
    <head></head><title></title>
<body bgcolor="beige">
    <p align="center"><font size="4" face="tahoma"><B>Ações do
Administrador:</B></font></p><hr>
    <p align="center"><A HREF="cadastrarorientador.html">Cadastrar Orientador</a></p>
<br>
<br>
    <a href="logout.jsp"><p align="center">Logout</p></a><br>
    <p align="center"><font size="3" face="tahoma"><B>Logado como: <%= login
%></B></font></p>
</body>
</html>
```

Principal.jsp

```
<%@ include file="controleSessao.jsp" %>
<%@ page import="java.util.*, br.farn.carla.*" %>

<html>
<body bgcolor=beige>
<font size=3 color=black><b>
<font size="4" face="tahoma"><p align="center"><B><font color="#800000">Orientadores
Cadastrados:</font></B></P></font>

<TABLE align="center" BORDER="1" WIDTH="80%">
```

```

<tr>
    <TD><p align="center"><font size="4" face="times new
roman"><B>Ref.</B></font></TD></p>
    <TD><p align="center"><font size="4" face="times new
roman"><B>Nome:</B></font></TD></p>
    <TD><p align="center"><font size="4" face="times new
roman"><B>Email:</B></font></TD></p>
    <TD><p align="center"><font size="4" face="times new
roman"><B>Projetos:</B></font></TD></p>
    <TD><p align="center"><font size="4" face="times new
roman"><B>Alterar</B></font></TD></p>
    <TD><p align="center"><font size="4" face="times new
roman"><B>Remover</B></font></TD></p>
</tr>
<%
ViewAdministrador adm = (ViewAdministrador) session.getAttribute("controleAdm");
if ( adm != null ) {
    Iterator list = adm.listaOrientadores();
    while ( list.hasNext() ) {
        OrientadorData orientador = (OrientadorData) list.next(); %>
        <tr>
            <td align="center"> <%= orientador.getId() %> </td>
            <td align="center"> <%= orientador.getNome() %> </td>
            <td align="center"> <%= orientador.getEmail() %> </td>
            <td align="center"> ----- </td>
            <td align="center"> <a href="alterarorientador.html">Alt</td>
            <td align="center"><A HREF="removerorientador.html"><IMG SRC="exc.gif"
        </A>
        </tr>
    }
}
%>
</TABLE>
</body>
</html>

```

ANEXO II**Códigos Java****ViewOrientador.java**

```
package br.farn.carla;

public class ViewOrientador {
    String nome;
    PersistenciaProjeto banco = new PersistenciaProjeto();

    public ViewOrientador() {
        nome = "default";
    }
}
```

ViewAdministrador.java

```
package br.farn.carla;

import java.util.*;

public class ViewAdministrador {
    PersistenciaProjeto banco = new PersistenciaProjeto();
    String nome;

    public ViewAdministrador() {
        nome = "default";
    }

    public ViewAdministrador(String login) {
        nome = login;
    }

    public Iterator listaOrientadores() {
        return banco.listaOrientadores().iterator();
    }
}
```

ProjetoPersistencia.java

```
package br.farn.carla;

import java.sql.*;
```

```

import java.util.*;

public class PersistenciaProjeto {
    private Connection con;
    private Statement instr;
    private ResultSet resultado;
    private String erro;

    public PersistenciaProjeto() {
        try {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            con = DriverManager.getConnection("jdbc:odbc:Projeto", "carla", "123");
        } catch ( ClassNotFoundException cnfex ) {
            erro = cnfex.getMessage();
        } catch ( SQLException sqllex ) {
            erro = sqllex.getMessage();
        }
    }

    public boolean autentica(String tipo, String login, String senha) {
        String consulta = "SELECT * FROM ";
        if ( tipo.equals("adm") )
            consulta += "Administrador ";
        else if ( tipo.equals("prof") )
            consulta += "Orientador ";
        else if ( tipo.equals("aluno") )
            consulta += "Orientando ";
        consulta += "WHERE login='"+login+"' and senha='"+senha+"'";
        System.out.println(consulta);
        System.out.println("Erro: "+erro);
        if ( con!=null ) {
            try {
                instr = con.createStatement();
                resultado = instr.executeQuery(consulta);
                boolean ok = resultado.next();
                instr.close();
                System.out.println(ok);
                return ok;
            } catch ( SQLException sqllex ) {
                erro = sqllex.getMessage();
                System.out.println("Erro: "+erro);
            }
        }
        return false;
    }
}

```

```

public Vector listaOrientadores() {
    Vector retorno = new Vector();
    String consulta = "SELECT * FROM Orientador";
    System.out.println("Erro: "+erro);
    if ( con!=null ) {
        try {
            instr = con.createStatement();
            resultado = instr.executeQuery(consulta);
            while ( resultado.next() ) {
                long id = resultado.getLong(1);
                String nome = resultado.getString(2);
                String email = resultado.getString(3);
                OrientadorData novo = new OrientadorData(id, nome, email);
                retorno.addElement(novo);
            }
            instr.close();
        } catch ( SQLException sqllex ) {
            erro = sqllex.getMessage();
            System.out.println("Erro: "+erro);
        }
    }
    return retorno;
}

```

ControleProjeto.java

```
package br.farn.carla;
```

```

public class ControleProjeto {
    PersistenciaProjeto banco;

    public ControleProjeto() {
        banco = new PersistenciaProjeto();
    }

    public ViewAdministrador autenticaAdm(String login, String senha) {
        if ( banco.autentica("adm",login, senha) )
            return new ViewAdministrador();
        return null;
    }
}

```