

INSTITUTO DE EDUCAÇÃO SUPERIOR DA PARAÍBA – IESP  
MIKEYAS ROCHA DA SILVA

A IMPORTÂNCIA DA APLICAÇÃO DE PADRÕES DE PROJETOS NO  
DESENVOLVIMENTO DE SOLUÇÕES: aplicação prática no desenvolvimento de  
agenda telefônica

João Pessoa

2017

MIKEYAS ROCHA DA SILVA

A IMPORTÂNCIA DA APLICAÇÃO DE PADRÕES DE PROJETOS NO  
DESENVOLVIMENTO DE SOLUÇÕES: aplicação prática no desenvolvimento de  
agenda telefônica

Trabalho de Conclusão do Curso apresentado ao  
Instituto de Educação Superior da Paraíba – IESP,  
curso Superior de Bacharelado em Sistemas de  
Informação, como pré-requisito para obtenção do  
grau de Bacharel em Sistemas de Informação, sob  
orientação da professora Msc. Alana Moraes.

João Pessoa

2017

Dedico este trabalho, primeiramente a Deus que sempre esteve presente em minha vida, me encorajando e dando forças para vencer, aos meus Irmãos que sempre me ajudaram a conquistar meus objetivos, a minha Esposa e Filho que são os meus maiores incentivos para vencer na vida, ao meu Pai que me educou para ser um homem íntegro, e especialmente a minha Mãe que sempre cuidou de mim, e fez o possível para me transformar em um vencedor. Sem sombra de dúvidas, Ela é a maior vencedora que eu já conheci em toda a minha vida, e todas as minhas conquistas eu devo a Ela.

## **AGRADECIMENTOS**

Agradeço a Deus acima de todas as coisas, pois sempre esteve comigo durante esta caminhada, e quando não pude caminhar, Ele me carregou em seus braços.

A minha esposa Ana Paula por todo apoio, paciência e estímulo que ela me deu ao longo dessa jornada, respeitando minhas ausências e me motivando.

A Professora e Orientadora Alana Moraes, pela valiosa dedicação na orientação deste trabalho, pela confiança e valorização de minhas potencialidades.

Aos demais professores do curso de Sistemas de Informação pelas contribuições acadêmicas, colaborando em minha formação como profissional e como ser humano.

A todos os amigos e colegas do curso pela convivência agradável, pelos conhecimentos compartilhados, e amizades cultivadas durante esses quase três anos de faculdade.

A Planc Engenharia e Incorporações LTDA, pela oportunidade de me conceder o estágio curricular.

Enfim, a todas as pessoas que, de maneira direta ou indireta, estiveram contribuindo para essa realização. Agradeço de coração.

## RESUMO

Os sistemas de informação vêm evoluindo com o passar do tempo, e estão se tornando cada vez mais complexos, não apenas em seus requisitos, mas também em sua implementação e manutenção. Isto faz com que os desenvolvedores utilizem todos os recursos necessários para garantir a escalabilidade, manutenibilidade e portabilidade desses sistemas. O objetivo deste trabalho é propor o desenvolvimento de uma aplicação web para gerenciamento de contatos telefônicos, aplicando as metodologias de padrões de projetos, para realizar a padronização do sistema, e solucionar algumas situações que comumente ocorrem em sistemas orientados a objeto. A utilização do processo de *software* em cascata como metodologia para o desenvolvimento do sistema, auxilia na realização de levantamento de requisitos, modelagem do sistema, implementação da aplicação, até sua implantação de fato. A aplicação dos padrões de projeto e arquiteturais garante um bom desenho da aplicação e também preserva seus benefícios no processo de manutenção. Para implementação do sistema, foram utilizadas as linguagens C# e ASP.NET no *backend*, e HTML5, CSS3 e JavaScript no *frontend*. Como resultados obtidos, observou-se a importância do planejamento do projeto e utilização de metodologias que auxiliam o planejamento e desenvolvimento do projeto; a necessidade de utilização de diagramas UML no planejamento do projeto; a aplicação dos Padrões de Projetos como técnica de padronização; a viabilidade em se construir aplicações com linguagem e banco de dados gratuitos; e finalmente, apontou soluções extensíveis a outras ferramentas utilizadas na organização.

Palavras-chave: Agenda telefônica. Aplicação web. Padrões de projeto. DAO. MVC. Baixo acoplamento. Alta coesão.

## **ABSTRACT**

Information systems have been expanding over time, and they are becoming increasingly complex, not only in their requirements, but also in their implementation and maintenance. This makes developers use all the necessary resources in order to ensure the scalability, maintainability, and portability of these systems. Thus, the objective of this work is to propose the development of a web application for telephone contact management, applying the methodologies of project patterns, to standardize the system, and to solve some situations that might commonly occur in object oriented systems. The use of the cascade software process as a methodology for the development of the system assists in attaining requirements survey, system modeling, implementation of the application, until its current implementation. The application of project and architectural patterns guarantees a good application design and also preserves its benefits in the maintenance process. For the system implementation, the C# and ASP.NET languages were used on the backend, and HTML5, CSS3 and JavaScript on the frontend. As results achieved, it was observed the relevance of the project planning and use of methodologies that help the planning and development of the project; the need to use UML diagrams in project planning; the application of the design patterns as a standardization technique; the viability of making applications with free language and database; and finally, it pointed out extensible solutions to other tools used in the organization.

Keywords: Phonebook. Web application. Design patterns. DAO. MVC. Low Coupling. High Cohesion.

## **LISTA DE ABREVIATURAS E SIGLAS**

CSS	-	Cascading Style Sheets
DAO	-	Data Access Object
HTML	-	Hypertext Markup Language
ID	-	Identificador
LINQ	-	Language Integrated Query
MVC	-	Model, View, Controller
Navbar	-	Barra de Navegação
OO	-	Orientação a Objetos
SGBD	-	Sistema de Gerenciamento de Banco de Dados
SQL	-	Structured Query Language
UML	-	Unified Modeling Language

## LISTA DE FIGURAS

Figura 01 – Diagrama do processo de software em cascata .....	22
Figura 02 – Diagrama de caso de uso .....	26
Figura 03 – Diagrama de classe .....	27
Figura 04 – Diagrama de pacote .....	28
Figura 05 – Diagrama de sequência: Acessar agenda .....	29
Figura 06 – Diagrama de sequência: Fazer login .....	30
Figura 07 – Diagrama de sequência: Adicionar contato .....	31
Figura 08 – Tela inicial do sistema .....	34
Figura 09 – Lista de contatos cadastrados no sistema .....	35
Figura 10 – Tela de detalhes do contato cadastrado no sistema .....	36
Figura 11 – Tela de login no sistema .....	37
Figura 12 – Tela inicial do sistema para usuários autenticados .....	38
Figura 13 – Tela gerência de usuários do sistema .....	38
Figura 14 – Tela adicionar novo contato ao sistema .....	39
Figura 15 – Tela editar contato do sistema .....	40
Figura 16 – Remover contato do sistema .....	40
Figura 17 – Altera senha do usuário .....	41
Figura 18 – Cadastrar um novo .....	42
Figura 19 – Editar um usuário existente .....	42
Figura 20 – Resetar senha de um usuário .....	43



## **LISTA DE QUADROS**

Quadro 01 – Tipos de Padrões de Projetos .....	19
Quadro 02 – Requisitos funcionais do sistema .....	25
Quadro 03 – Requisitos não-funcionais do sistema .....	25

## SUMÁRIO

1. INTRODUÇÃO.....	10
1.1 Justificativa.....	11
1.2. Objetivo Geral.....	11
1.3. Objetivos Específicos.....	11
1.4. Estrutura do documento.....	12
2. FUNDAMENTAÇÃO TEÓRICA.....	13
2.1. A linguagem HTML.....	13
2.2. A linguagem Javascript.....	14
2.3. A linguagem C#.....	14
2.4. Banco de Dados MySQL.....	16
2.4.1. Características Principais.....	16
2.5. Padrões de Projetos – Conceitos.....	17
2.5.1. Vantagens de utilizar <i>Padrões de Projetos</i> .....	17
2.5.2. Tipos de <i>Padrões de Projetos</i> .....	18
2.5.3. <i>Padrão DAO</i> .....	19
2.5.4. <i>Padrão MVC</i> .....	20
2.5.5. <i>Padrão Baixo Acoplamento</i> .....	20
2.5.6. <i>Padrão Alta Coesão</i> .....	21
3. DESENVOLVIMENTO DO PROJETO.....	22
3.1. PROCESSO DE SOFTWARE.....	22
3.2. Materiais e Métodos.....	23
3.3. Levantamento de requisitos.....	23
3.4. Modelagem UML do sistema.....	26
3.5. Implementação do sistema.....	31
3.5.1. Aplicação do padrão MVC.....	32
3.5.2. Aplicação do padrão DAO.....	32
3.5.3. Aplicação do padrão Baixo Acoplamento.....	33
3.5.4. Aplicação do padrão Alta Coesão.....	33
3.6. Integração de sistema.....	34
3.7. Operação e manutenção.....	44
4. CONCLUSÃO.....	45
REFERÊNCIAS.....	47

## 1. INTRODUÇÃO

Este trabalho tem por objetivo estudar a importância da aplicação de padrões de projetos no desenvolvimento de uma aplicação web para armazenamento de contatos telefônicos (agenda telefônica) na empresa Planc Engenharia e Incorporações Ltda.

Os Padrões de Projetos, segundo Gamma *et al.* (2005), descrevem soluções para problemas recorrentes no desenvolvimento de sistemas de *software* orientados a objetos. Um padrão de projeto estabelece um nome e define o problema, a solução, quando aplicar esta solução e suas consequências. Os Padrões de Projetos têm como objetivo principal facilitar a reutilização de soluções de projeto, isto é, soluções na fase de projeto do software, sem considerar reutilização de código. Segundo Gamma *et al.* (2005), um padrão de projeto nomeia, abstrai e identifica os aspectos-chave de uma estrutura de projeto comum para torná-la útil para a criação de um projeto orientado a objeto reutilizável. Além disto, estes podem resultar em um vocabulário comum de projeto, facilitando comunicação, documentação e aprendizado dos sistemas de software.

Muitos dos documentos criados durante um processo de desenvolvimento de *software* são comuns a vários sistemas. Algumas vezes as mesmas técnicas são utilizadas para a criação desses documentos e, portanto, os mesmos problemas são encontrados. Os padrões apresentam uma forma de descrever soluções para esses problemas comuns baseado na experiência de outras pessoas.

Desta maneira, foram utilizados métodos para verificar a viabilidade de aplicação dos padrões de projetos no desenvolvimento de uma agenda telefônica de fácil utilização pelos usuários, de forma ágil e com menores contratempos. Atualmente, o armazenamento dos contatos telefônicos da empresa é feito em planilha de Excel, o qual impossibilita a simultaneidade do acesso aos dados da planilha, ficando restrita a apenas um usuário, que costumeiramente é a recepcionista da empresa.

Não foram encontradas ferramentas *open source* disponíveis no mercado, que atendessem com excelência a necessidade da organização. Então, foi proposto desenvolver uma solução que atendesse tal necessidade.

Com a conclusão da pesquisa, verificou-se que o sistema proposto contribuiu para a melhoria no sistema de armazenamento de contatos telefônicos dos clientes e colaboradores da organização. A pesquisa mostrou que a situação atual do sistema utilizado pela empresa, e a real necessidade da mesma. Desta forma, foi possível atender, em sua totalidade, as necessidades apresentadas pela empresa.

## 1.1 JUSTIFICATIVA

Constantemente, os colaboradores da organização precisam entrar em contato com os clientes, fornecedores e prestadores de serviços de uma forma geral. Contudo, tal atividade enfrenta dificuldades com as limitações da atual ferramenta, tornando constantemente difícil o acesso aos contatos telefônicos. A ferramenta atual é executada apenas em uma máquina (recepção) e exige o auxílio da recepcionista durante a busca do contato desejado pelos colaboradores. Além disto, uma problemática relevante nesta estrutura é o extravio de contatos. Neste sentido, é necessária a pesquisa e desenvolvimento de uma alternativa para o sistema atual, que facilite o acesso de todos, não possua intervenção de terceiros e garanta que o armazenamento dos contatos seja assegurado.

## 1.2. OBJETIVO GERAL

O objetivo geral deste trabalho é desenvolver uma aplicação web para armazenamento de contatos telefônicos (sistema de gerenciamento de contatos telefônicos) na empresa Planc Engenharia e Incorporações Ltda. Para atingir tal objetivo, este documento delineou uma série de objetivos específicos, foram eles:

## 1.3. OBJETIVOS ESPECÍFICOS

- Avaliar os diagramas de caso de uso, de classe, e os requisitos funcionais e não funcionais para o sistema proposto;
- Apresentar os padrões de projetos que melhor se aplicam ao sistema proposto;
- Explanar a reutilização de soluções de projeto, isto é, soluções na fase de projeto do software, sem considerar reutilização de código;
- Aplicar padrões de projetos no desenvolvimento da aplicação web;

#### 1.4. ESTRUTURA DO DOCUMENTO

Para discutir a ferramenta proposta, o presente documento foi organizado em 4 seções. A seção 1 discutiu as informações iniciais sobre o problema, objetivos da pesquisa e as etapas percorridas. Neste sentido, a seção 2 apresentou conceitos teóricos relevantes para a construção do sistema proposto, as linguagens, banco de dados e tecnologias utilizadas, os conceitos de padrões de projetos, seus tipos, quais as vantagens em utilizá-los, e os padrões utilizados no desenvolvimento do projeto. A seção 3 tratou do desenvolvimento do projeto, explicando o processo de software em cascata, materiais e métodos utilizados para desenvolvimento do projeto, o levantamento de requisitos do sistema, a modelagem UML do sistema, com a apresentação dos diagramas UMLs, e por fim, detalhou as etapas de implementação do sistema, integração, e operação e manutenção do sistema. Por fim, a seção 4 apresentou a conclusão do trabalho, os principais resultados obtidos com a implementação e implantação do sistema na organização.

## 2. FUNDAMENTAÇÃO TEÓRICA

A pesquisa foi norteada pela busca de soluções de baixo custo, que possibilitassem uma fácil implementação e manutenção, além de servir como direcionador no desenvolvimento de ferramenta proposta para gerenciamento de contatos telefônicos. Como linguagens, optou-se por HTML5, com CSS3, e Javascript, para desenvolvimento *frontend*, e C# e ASP.NET, para desenvolvimento *backend*, por serem gratuitas, de fácil entendimento e por possibilitarem o uso de Padrões de Projetos para a padronização e o melhor aproveitamento. Para realização da gestão da base de dados, utilizamos o MySQL, pois é uma ferramenta *Open Source* de grande aceitação no mercado.

As tecnologias utilizadas para desenvolvimento do sistema proposto neste trabalho, foram escolhidas por serem gratuitas, por serem bem-aceitas pela comunidade, por ter vasta documentação disponível na web, e também porque a equipe de desenvolvimento já dispõe de conhecimentos relacionados a elas, o que evita a necessidade de ter que aprender uma nova tecnologia do zero, podendo direcionar os esforços para o objetivo principal deste trabalho.

### 2.1. A LINGUAGEM HTML

Segundo Silva (2008, p. 26), “HTML é a sigla em inglês para *HyperText Markup Language*, que, em português, significa linguagem para marcação de hipertexto”, e é a linguagem interpretada pelos navegadores para exibição de conteúdo. Hipertexto pode ser definido como todo conteúdo inserido em um documento para web, onde sua principal característica é se interligar a outros documentos da web.

A linguagem HTML foi criada inicialmente para exibir documentos científicos, para possibilitar que cientistas do mundo inteiro compartilhassem eletronicamente seus textos e pesquisas. Porém, com o tempo e a evolução da *web* e seu potencial comercial, tornou-se necessária a exibição de informações com grande riqueza de elementos gráficos e de interação. O HTML é um conjunto de *tags* responsáveis pela marcação do conteúdo de uma página no navegador. Diversas *tags* são disponibilizadas pela linguagem HTML e cada uma possui uma funcionalidade específica (SILVA, 2008).

Silva (2010) afirmou que não há como fazer funcionar um formulário HTML com o uso de elementos HTML, pois ela apenas cria os rótulos e campos de um formulário para serem preenchidos pelo usuário e nada mais. Com HTML, não se consegue processar os dados

nem mesmo enviá-los ao servidor ou a outra máquina qualquer. Para isso, é necessário utilizar um programa que consiga manipular e processar os dados.

## 2.2. A LINGUAGEM JAVASCRIPT

JavaScript é a linguagem de programação mais popular no desenvolvimento *web*. Suportada por todos os navegadores, a linguagem é responsável por praticamente qualquer tipo de dinamismo que queiramos em nossas páginas. O JavaScript é uma linguagem de *scripting*, este tipo de linguagem geralmente é definido como uma linguagem de programação que permite ao programador controlar uma ou mais aplicações de terceiros. Com o JavaScript, é possível controlar alguns comportamentos dos navegadores por meio de trechos de código enviados na página HTML. As linguagens *scripting* são linguagens interpretadas, e não dependem do processo de compilação para serem executadas (SILVA, 2010).

Silva (2010, p. 20), diz que o “Javascript foi criada pela Netscape em parceria com a Sun Microsystems, com a finalidade de fornecer um meio de adicionar interatividade a uma página web”. Silva (2010), ainda afirma que Javascript é uma linguagem desenvolvida para rodar no lado do cliente, isto é, a interpretação e o funcionamento da linguagem depende de funcionalidades hospedadas no navegador do usuário. Isso é possível porque existe um interpretador javascript hospedado no navegador.

O JavaScript também é tolerante a erros, pois a linguagem realiza conversões automáticas durante operações. Mas algumas dessas conversões podem resultar em *bugs* no sistema (SILVA, 2010).

## 2.3. A LINGUAGEM C#

Para alguns autores, A linguagem C# é elegante e de tipos protegidos, orientada a objeto e que permite aos desenvolvedores construírem uma variedade de aplicações seguras e robustas, compatíveis com o .NET Framework. Dentre suas características essenciais, podemos citar a simplicidade, pois apesar de ser uma linguagem poderosa, é simples de se usar, ela é completamente orientada a objetos, fortemente tipada, o que ajuda a evitar erros por manipulação imprópria de tipos, ela possibilita controle de versões, tem suporte a código legado, gera código gerenciável, etc.

Com o C# é possível criar aplicações tradicionais do Windows, Web services baseados em XML, componentes distribuídos, aplicativos cliente-servidor, aplicativos com banco de dados e muito, muito mais. O Microsoft Visual Studio é um editor de códigos avançado que fornece modelagem de interface do usuário conveniente, depurador integrado, e muitas outras ferramentas para facilitar o desenvolvimento de aplicativos na linguagem C#, utilizando o .NET Framework (MICROSOFT, 2012; LIMA, 2002).

Como uma linguagem orientada a objetos, o C# suporta os conceitos de encapsulamento, herança e polimorfismo. Todas as variáveis e métodos, incluindo o método principal (*Main*), o ponto de execução de uma aplicação, são encapsuladas em definições de classes. Uma classe derivada pode herdar diretamente somente de uma classe pai, mas pode herdar de qualquer quantidade de interfaces. Métodos da classe derivada que substituem métodos virtuais de uma classe pai exigem a utilização da palavra-chave *override* como forma de evitar a redefinição acidental (MICROSOFT, 2012).

Microsoft (2012) afirmou que além desses princípios básicos orientados a objeto, C# torna fácil desenvolver componentes de *software* através de várias construções de linguagem inovadoras, incluindo o seguinte:

- Assinaturas de métodos encapsulados, chamadas *delegates*, que permitem notificações de evento de tipo seguro;
- Propriedades, que servem como acessadores para variáveis de membro particular;
- Atributos, que fornecem metadados declarativos sobre tipos em tempo de execução;
- *In-line* comentários de documentação XML;
- LINQ (Consulta Integrada à Linguagem) que fornece recursos internos de consulta por uma variedade de fontes de dados.

O processo de compilação do C# é simples comparado com C++ e mais flexível que em Java. Não há arquivos de cabeçalho separados, e não há a necessidade de que métodos e tipos sejam declarados em uma ordem específica. Um arquivo de código em C# pode definir qualquer número de classes, estruturas, interfaces e eventos (LIMA, 2002).



## 2.4. BANCO DE DADOS MYSQL

Conforme o Manual de Referência Mysql (2010), o Mysql é um sistema de gerenciamento de banco de dados (SGBD) relacional, que utiliza a linguagem SQL (*Structured Query Language* – Linguagem de Consulta Estruturada) como interface, e é um *software Open Source*. A intenção inicial era de usar o mSQL para conectar às tabelas utilizando rápidas rotinas de baixo nível (ISAM). Mas após a realização de alguns testes, verificou-se que o mSQL não era rápido e nem flexível o suficiente para nossas necessidades. Isto resultou em uma nova interface SQL para o banco de dados, mas com praticamente a mesma Interface API do mSQL. Esta API foi escolhida para facilitar a portabilidade para códigos de terceiros que era escrito para uso com mSQL para ser portado facilmente para uso com o Mysql.

Milani (2006) afirma que o Mysql é atualmente um dos bancos de dados mais populares do mundo, foi criado na Suécia por dois suecos e um finlandês: David Axmark, Allan Larsson e Michael "Monty" Widenius, que trabalharam juntos desde a década de 1980. Para Milani (2006), o sucesso do MySQL deveu-se à fácil integração com o PHP incluído, quase que obrigatoriamente, nos pacotes de hospedagem de sites da Internet oferecidos atualmente.

### 2.4.1. Características Principais

Conforme o Mysql (2010), em seu Manual de Referência, segue abaixo algumas das características mais importantes do MySQL:

- Portabilidade (suporta praticamente qualquer plataforma atual);
- Compatibilidade (existem *drivers* e módulos de interface para diversas linguagens de programação);
- Excelente desempenho e estabilidade;
- Pouco exigente quanto a recursos de hardware;
- Facilidade de uso;
- É um *software* gratuito;
- Suporte a vários tipos de tabelas (como MyISAM e InnoDB), cada um destinado a um tipo de aplicação;

- Faltam alguns recursos (Triggers por exemplo) quando comparados como outros bancos de dados, como o PostgreSQL.

## 2.5. PADRÕES DE PROJETOS – CONCEITOS

Os Padrões de Projetos capturam soluções desenvolvidas e evoluídas com o passar do tempo e que já funcionaram, e que por esse motivo devem ser reutilizadas (reuso de experiências anteriores). Um padrão descreve soluções para situações que ocorrem em seu ambiente, e estas soluções podem ser usadas diversas vezes, sem nunca fazer a mesma coisa repetida. É a reutilização de projetos e arquiteturas de sucesso, ou seja, técnicas comprovadas, em forma de um catálogo, em um formato consistente e acessível para projetistas, onde cada padrão de projeto sistematicamente nomeia, explica, e avalia um projeto importante que ocorre várias vezes em sistemas OO. O padrão de projeto descreve como os objetos se comunicam sem interferir com modelos de dados e métodos de outros objetos. São soluções exaustivamente refinadas, resultado de um longo processo, testes e reflexão sobre o que torna um sistema mais flexível, reusável, modulado e compreensível, construídas em grupo utilizando um vocabulário comum, capturam, de forma sucinta e facilmente aplicável, soluções do projeto que foram desenvolvidas e evoluíram com o passar do tempo (FREEMAN, 2009).

### 2.5.1. Vantagens de utilizar Padrões de Projetos

Segundo Gamma (2005), algumas das vantagens que se pode obter estudando e aplicando os *Padrões de Projetos*:

- **Padronização:** Um padrão é a documentação de um problema e sua solução, e os Padrões de Projetos representam padrões pré-definido. Ao utilizar estes padrões, o código fica mais organizado, o que facilita a comunicação entre a equipe, podendo melhorar a documentação, padronizar a nomenclatura de classes, métodos e propriedades, entre outras vantagens.
- **Facilitar as atividades de projeto:** O planejamento do projeto fica mais simplificado com o uso de soluções que foram definidas anteriormente, para problemas comuns.

- **Redução de acoplamento (dependências):** Com o uso de alguns Padrões de Projetos é possível reduzir a quantidade de dependências entre os objetos, o que aumenta a flexibilidade e a reutilização de código, pois um objeto sem muitas dependências se torna muito mais flexível.
- **Facilitar modificações:** Com a utilização de Padrões de Projetos pode-se facilitar as modificações nas funcionalidades de um aplicativo. É possível realizar alterações em classes e subsistemas sem que afete os seus clientes. Estas facilidades podem aumentar a flexibilidade do aplicativo, para que se adapte aos novos requisitos.
- **Alternativas à herança:** Com o uso de Padrões de Projetos se tem algumas alternativas ao uso de herança, e tais alternativas têm o objetivo de oferecer maior flexibilidade ao funcionamento da aplicação. A herança é uma forma de se reutilizar o código, que pode agregar vantagens e desvantagens ao projeto.
- **Distribuir responsabilidades:** O uso de Padrões de Projetos auxilia na distribuição de responsabilidade entre os objetos da aplicação, o que pode maximizar o potencial de reutilização de código.
- **Acelerar o crescimento:** Os Padrões de Projetos são criados através de pesquisas nos trabalhos realizados por programadores experientes, e sua utilização pode auxiliar um programador menos experiente em orientação a objetos na utilização de práticas características de programadores mais experientes, e seus conhecimentos acumulados. Concomitadamente, o estudo de Padrões de Projetos também pode ser muito útil para programadores experientes.

### 2.5.2. Tipos de Padrões de Projetos

Segundo Gamma (2005), os Padrões de Projetos foram organizados segundo as suas funções e os principais tipos foram elencados a seguir:

Quadro 01 – Tipos de Padrões de Projetos

<b>Tipo</b>	<b>Descrição</b>
Criacionais	São aqueles que criam objetos (em vez de termos que instanciá-los diretamente). Dão ao programa mais flexibilidade na decisão de quais objetos precisam ser criados para um determinado caso. Tornam um sistema independente de como seus objetos são criados, compostos e representados. Todos os padrões criacionais tratam da melhor maneira como instanciar objetos. É importante porque o programa não depende de como os objetos são criados ou organizados. Em muitos casos a natureza exata de um objeto que é criado pode variar de acordo com as necessidades do programa e abstrair o processo de criação em uma “Classe Criadora” especial pode tornar o programa mais flexível e geral.
Comportamentais	Ajudam a definir a comunicação entre objetos no sistema e como o seu fluxo é controlado em um programa complexo. Tratam de algoritmos e como atribuir responsabilidades entre objetos.
Estruturais	Ajudam a compor grupos de objetos em grandes estruturas como interfaces complexas ou contagem de dados. Tratam de compor classes e objetos para formar estruturas grandes e complexas.

*Fonte: GAMMA, 2005, p. 28.*

### 2.5.3. Padrão DAO

Segundo Trindade e Fisher (2007), o principal objetivo do padrão DAO é abstrair da camada de negócios, o mecanismo de persistência de dados utilizado na aplicação. Sardagna e Vahldick (2007, p. 2), afirmaram que com a aplicação do padrão DAO, “a camada de negócios acessa os dados persistidos, sem ter conhecimento se os dados estão em um banco relacional ou em um arquivo XML.”

Trindade e Fisher (2007) afirmou que este padrão de projeto pode ser utilizado em aplicações que façam acesso a qualquer tipo de fonte de dados, ou caso haja a necessidade de acessar outros tipos quaisquer durante ou após o desenvolvimento da aplicação.

#### 2.5.4. Padrão MVC

Segundo Liberty e Horovitz (2009), o padrão Model-View-Controller é um padrão de criação que foi desenvolvido por Trygve Mikkjel Heyerdahl Reenskaug, quando trabalhava no Smalltalk em 1979, na Xerox PARC.

Para Liberty e Horovitz (2009), os três principais conceitos do padrão MVC são: Primeiro: O projeto é iniciado com um modelo, ou seja, uma representação do domínio do problema. Este modelo possui o estado da aplicação e seus dados, e se concentra na estrutura e na manipulação dos dados. Segundo: O modelo é apresentado para o usuário através da visualização. Esta geralmente inclui controles com os quais o usuário interage. Terceiro: O controlador responde as requisições do usuário, e intermedia as interações entre o modelo e a visualização, podendo modificar um ou outro.

Liberty e Horovitz (2009), afirmou que a Microsoft fornece um MVC Framework para ASP.NET, que realiza o mapeamento do padrão de criação MVC para o .NET Framework, adicionando modelos que facilitam a criação de aplicações web MVC.

#### 2.5.5. Padrão Baixo Acoplamento

Larman (2007, p. 314) explicou que acoplamento “é a medida de quão forte um elemento está conectado a, tem conhecimento de, ou depende de outros elementos”. Ou seja, um elemento com um acoplamento baixo não depende de muitos outros elementos. Esses elementos incluem classes, subsistemas, sistemas, etc. Quando uma classe tem um acoplamento forte, ela depende de muitas outras classes, e podem ser mais difíceis de se entender isoladamente; podem ser mais difíceis de reutilizar, pois para utilizá-las é necessário a presença adicional das classes das quais é dependente; ou pode sofrer modificações forçadas, devido a modificações em classes relacionadas.

Larman (2007, p. 315), disse que o “Acoplamento baixo é um princípio a se ter em mente durante todas as decisões de projeto”, e deve ser considerado continuamente. É um princípio de avaliação que deve ser aplicado em todas as decisões de projeto.

A aplicação do padrão Baixo Acoplamento orienta atribuir responsabilidades de forma que essa atribuição não aumente o acoplamento há um nível que leve a resultados negativos. Acoplamento baixo reduz o impacto de modificações, melhorando o projeto de classes independentes. Este padrão está relacionado a outros padrões, como Coesão

Alta, e precisa ser incluído entre os princípios de projeto que influenciam as escolhas na atribuição de responsabilidade (LARMAN, 2007).

#### *2.5.6. Padrão Alta Coesão*

Larman (2007, p. 315) explicou que a coesão “é uma medida de quanto as responsabilidades de um elemento estão fortemente relacionadas e focalizadas”. Quando um elemento possui responsabilidades relacionadas, e não executa um grande volume de trabalho, este tem coesão alta.

Larman (2007, p. 315), disse que “assim como o Acoplamento Baixo, a Coesão Alta é um princípio que devemos ter em mente durante todas as decisões de projeto”, e deve ser considerado continuamente. É um princípio de avaliação que deve ser aplicado em todas as decisões de projeto. Os projetistas aplicam este princípio de avaliação em todas as decisões de projeto.

As vantagens de se ter classes com coesão alta é que ela se torna relativamente fácil de se manter, compreender e reutilizar. Fazendo uma analogia entre o padrão Coesão Alta e o mundo real, é que se uma pessoa assume muitas responsabilidades não relacionadas, principalmente se essas responsabilidades deveriam ser delegadas a outras pessoas, então esta pessoa não é eficiente.

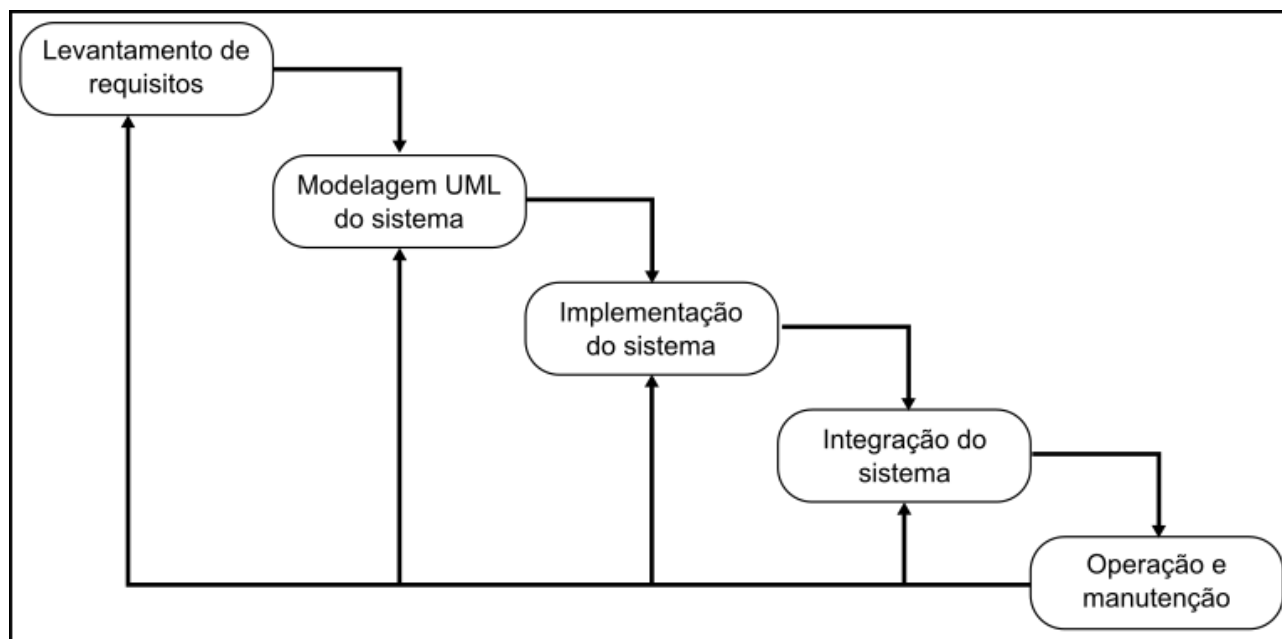
### 3. DESENVOLVIMENTO DO PROJETO

De acordo com as necessidades impostas pelo ambiente e buscando alternativas viáveis para o desenvolvimento do sistema *web*, optou-se pela utilização de Padrões de Projetos para padronizar o desenvolvimento em nível de projeto, pela linguagem C# e Asp.NET por serem gratuitas, flexíveis e relativamente fáceis de se programar, e possibilitar a aplicação dos paradigmas de OO (Orientação a Objeto) e como servidor de banco de dados foi utilizado o MySQL, por ser um SGBD gratuito.

#### 3.1. PROCESSO DE SOFTWARE

Neste projeto de software, optou-se por utilizar o modelo de processo de software em cascata ou ciclo de vida de software. Este modelo levou em consideração as atividades fundamentais do processo de especificação, desenvolvimento, validação e evolução. Além disto, tais fases foram trabalhadas de modo distinto, de acordo com as fases do modelo em cascata que refletem diretamente as atividades fundamentais do processo de desenvolvimento de software (SOMMERVILLE, 2011). A figura 01 mostrou as fases do processo de software deste projeto:

Figura 01 – Diagrama do processo de software em cascata.



*Fonte: Própria autoria.*

A escolha do modelo em cascata se deu porque os requisitos do sistema foram bem compreendidos e provavelmente não sofrerão nenhuma mudança radical no decorrer do desenvolvimento do sistema.

### 3.2. MATERIAIS E MÉTODOS

A pesquisa foi executada por meio de levantamento de requisitos, junto à gerência da organização, com a procura de tecnologias e métodos adequados para a construção de uma solução dentro das possibilidades administrativas da organização. Para isso foi feito uma revisão bibliográfica e levantamento de requisitos.

Para o desenvolvimento foram utilizados dois computadores, sendo um utilizado como servidor e outro como cliente para realização dos testes necessários e também, *softwares* de apoio já existentes no mercado (linguagens de programação apropriada para a criação de *sistemas web* com a aplicação de padrões).

Foram utilizadas as seguintes ferramentas de apoio ao desenvolvimento:

- MS Visual Stúdio 2017 Community - (<https://www.visualstudio.com/pt-br/downloads/>);
- Astah Community v7.1.0 - (<http://astah.net/download>);
- MySQL Server - (<https://dev.mysql.com/downloads/>);

### 3.3. LEVANTAMENTO DE REQUISITOS

O levantamento de requisitos foi uma das primeiras fases de um processo de desenvolvimento de software. Segundo Guedes (2011, p. 20), “esta fase trabalha com o domínio do problema e tenta determinar ‘o que’ o software deve fazer e se é realmente possível desenvolver o software solicitado”. Nesta etapa buscou-se entender a real necessidade do usuário e o que ele deseja que o sistema realize.

Foi identificado que a atual ferramenta de gerenciamento dos contatos telefônicos utilizada pela empresa é uma planilha de dados em MS Excel, onde estão inseridos o nome, telefone e observação de cada contato telefônico. Esta planilha fica armazenada no computador da recepção da empresa, e é alimentada pela recepcionista. Quando algum outro colaborador necessita de algum contato telefônico contido na ferramenta, este é solicitado a recepcionista, que busca na planilha, e retorna a informação ao solicitante, caso a mesma seja encontrada. Sendo assim, foi estudado a



viabilidade de implantação de um novo sistema web de gerenciamento de contatos telefônicos, que facilite o acesso dos colaboradores aos contatos armazenados. Este sistema poderá ser acessado pela rede interna da empresa, através do navegador, em qualquer terminal ligado a rede corporativa.

Em entrevista com o responsável pela Gerência Administrativa da organização, foram obtidas informações relevantes para o trabalho. Foi perguntado “Quais são as dificuldades encontradas no acesso à atual ferramenta de gerenciamento de contatos telefônicos, para inclusão, alteração, exclusão e consulta dos dados contidos nesta?”, a resposta foi a seguinte:

A atual ferramenta não é confiável, pois seu acesso é apenas local, no terminal da recepção. Sua inclusão fica restrita ao usuário do terminal da recepção (recepcionista), e nem sempre os contatos são armazenados da forma correta. Também temos dificuldades no acesso aos contatos telefônicos armazenados, pois como o acesso é local, sempre temos que solicitá-lo ao usuário do terminal, que por sua vez, precisa parar sua atividade para realizar a consulta dos dados na ferramenta. Podemos apontar ainda que a atual ferramenta não possui nenhum mecanismo anti-falhas, de forma que se o usuário acidentalmente pressionar o delete, os dados da planilha serão deletados. Já perdemos vários contatos desta maneira.

Também foi perguntado “O que espera que a ferramenta de gerenciamento de contatos telefônicos faça?”:

Esperamos que seja um sistema simplificado, para que cada departamento, e cada colaborador com acesso a um terminal conectado à rede corporativa da organização possa acessar o sistema de gerenciamento de contatos telefônicos, para consultar os dados de forma simplificada, sem a necessidade de acionar a recepcionista. Também desejamos que os dados inseridos no sistema fiquem protegidos de falhas operacionais, e que apenas usuários logados consigam inserir, editar ou excluir os dados no sistema.

O levantamento de requisitos identifica dois tipos de requisitos: os funcionais e os não-funcionais. Os requisitos funcionais correspondem ao que o cliente quer que o sistema realize, ou seja, as funcionalidades do software. Os requisitos não-funcionais correspondem às restrições, condições, consistências, validações baseadas nos requisitos funcionais do sistema.

Desta forma, por meio das informações obtidas com a aplicação do questionário, foram identificados os requisitos funcionais e não funcionais do sistema proposto, organizados nas tabelas que se seguem:

Quadro 02 – Requisitos funcionais do sistema

ID	Descrição
RF01	O sistema deverá fazer o armazenamento dos contatos telefônicos de colaboradores e fornecedores da organização;
RF02	O sistema deverá ser acessado por todos os terminais conectados a rede da empresa, sem necessidade de instalação de aplicações;
RF03	Não será necessário que o usuário esteja logado no sistema, para realizar consultas à base de contatos telefônicos do sistema;
RF04	O sistema deverá possibilitar a inclusão de novos contatos em sua base de dados;
RF05	Cada contato do sistema deverá conter nome, telefone e um campo de lembrança sobre o contato;
RF06	O sistema deverá permitir que apenas usuários logados façam a manutenção do sistema (inclusão, exclusão e alteração dos dados);
RF07	O sistema deverá permitir que apenas o administrador consiga adicionar novos usuários ao sistema;

*Fonte: Própria autoria.*

Quadro 03 – Requisitos não-funcionais do sistema

ID	Descrição
RNF01	O sistema deverá ser executado em navegadores web;
RNF02	O sistema deverá ser implantado em um servidor, e ser acessado pelas máquinas clientes através da rede da empresa;
RNF03	O sistema deverá estar disponível durante todo o expediente da empresa;
RNF04	O processo de consulta do sistema poderá ser realizado por mais de um usuário simultaneamente;
RNF05	Os dados dos contatos inseridos no sistema deverão ser inseridos em um banco de dados;

*Fonte: Própria autoria.*

Conforme pode-se observar nas tabelas 02 e 03, durante a realização do levantamento de requisitos, foram identificados sete requisitos funcionais e cinco requisitos não funcionais para o sistema proposto.

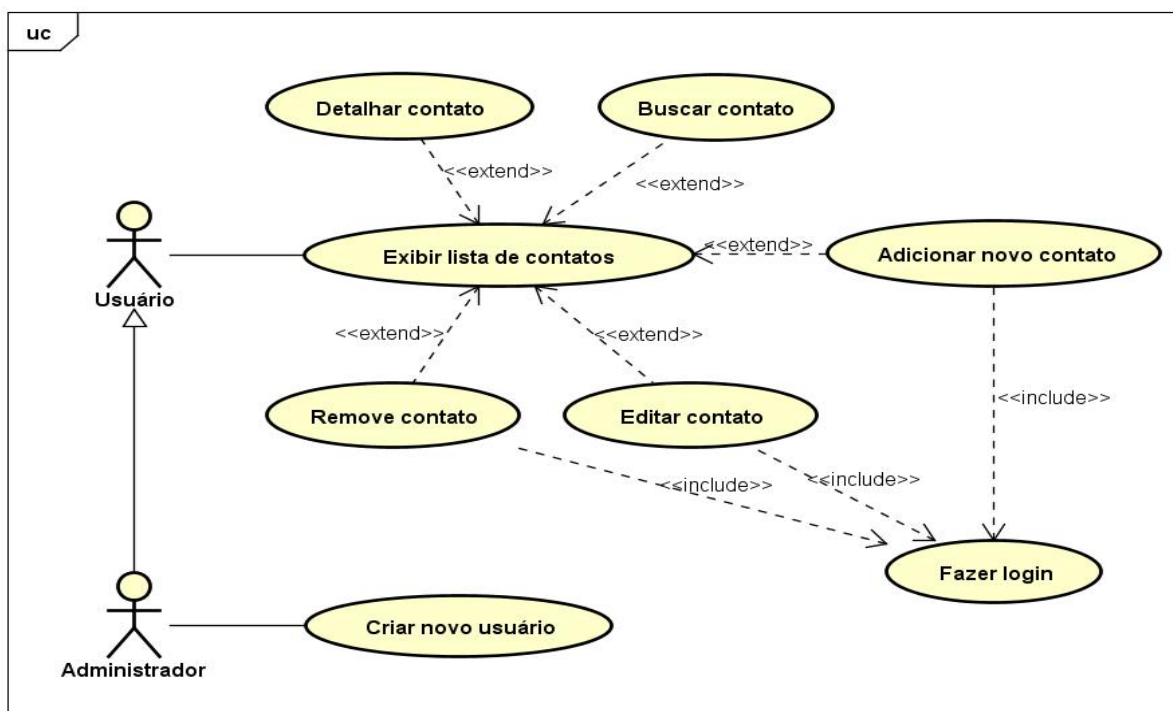
### 3.4. MODELAGEM UML DO SISTEMA

Segundo Guedes (2011, p. 20), “[...] todo e qualquer sistema deve ser modelado antes de se iniciar sua implementação, entre outras coisas, porque os sistemas de informação frequentemente costumam ter a propriedade de ‘crescer’”. Para Guedes (2011), os sistemas de informação estão constantemente mudando, pois eles sofrem influência dos clientes, que sempre querem modificá-los, das mudanças constantes de mercado, das leis e diretrizes governamentais, entre outras. Guedes (2011) ainda disse que uma das vantagens de se modelar um sistema é que esta é uma forma bastante eficiente de documentá-lo.

Um modelo de software é uma abstração do sistema, descrevendo aspectos estruturais ou comportamentais do mesmo, onde apenas é incluído no modelo os aspectos do sistema físico, relevantes ao modelo. Sendo assim, em um diagrama de caso de uso serão exibidos os requisitos necessários ao sistema, identificando as funcionalidades do software e os atores que poderão utilizá-las.

Cada tipo de diagrama UML analisa o sistema, ou parte dele, conforme uma visão específica, onde alguns diagramas enfocam o sistema de forma mais geral, como é o caso do diagrama de caso de uso.

Figura 02 – Diagrama de caso de uso

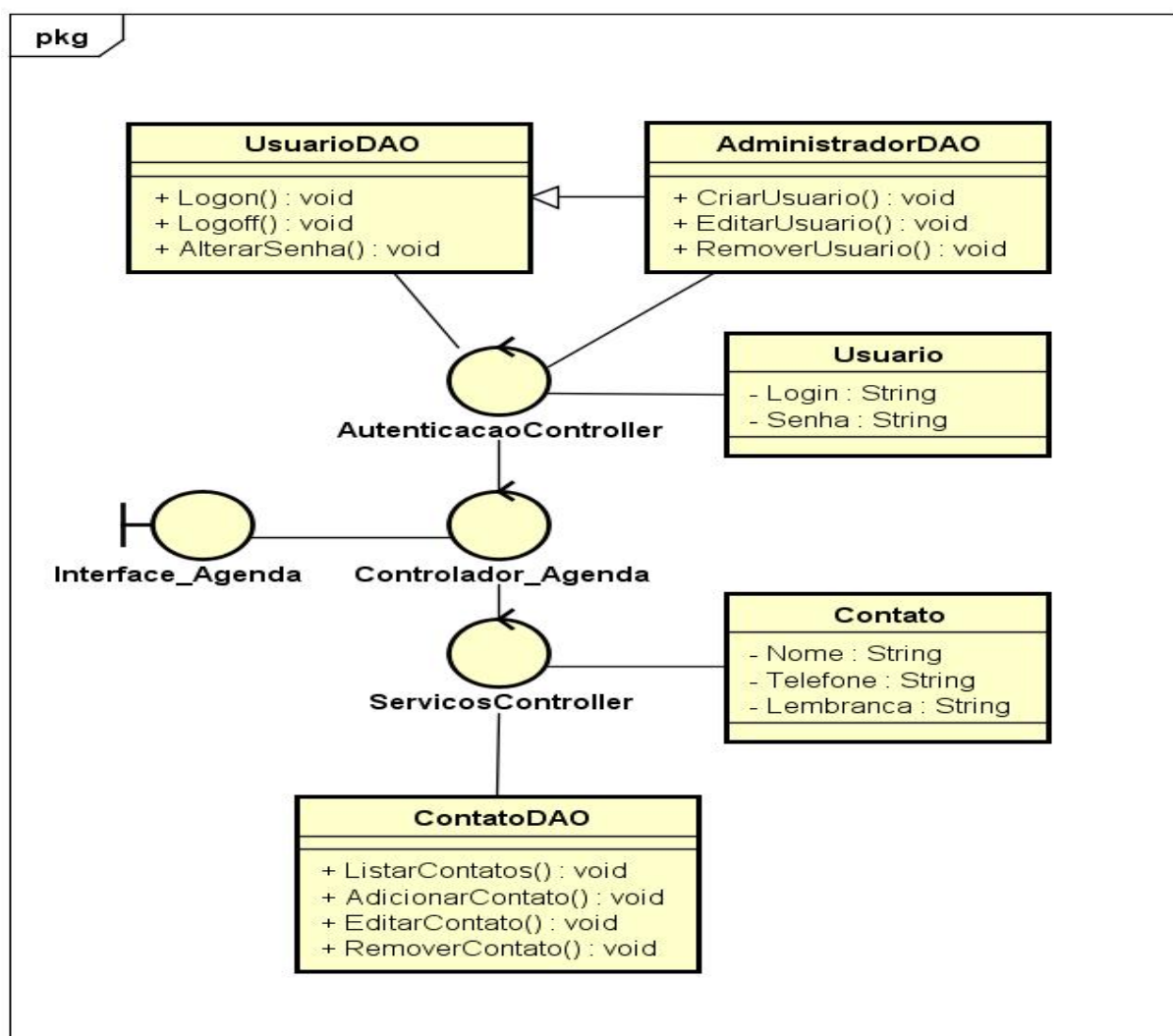


Fonte: Própria autoria.

No diagrama de caso de uso a seguir (figura 02), exibe as funcionalidades do sistema e os seus atores. Dois atores podem interagir com o sistema: o usuário e o administrador. O usuário consegue exibir a lista de contatos, buscar contatos na lista e acessar detalhes dos contatos, sem a necessidade de realizar o login no sistema. Já as funcionalidades de adicionar, editar ou remover contatos do sistema, exige que o usuário realize o login no sistema. O administrador herda todas as funcionalidades que o usuário pode executar, mas apenas o administrador consegue incluir um novo usuário do sistema, e essa é a principal diferença entre o usuário e o administrador.

O diagrama de classe (figura 03) é o mais importante da UML, pois ele apoia a maioria dos demais diagramas. Sua função é definir a estrutura das classes utilizadas pelo sistema, determinando os atributos e métodos que cada classe tem, estabelecendo os relacionamentos e comunicação entre elas.

Figura 03 – Diagrama de classe

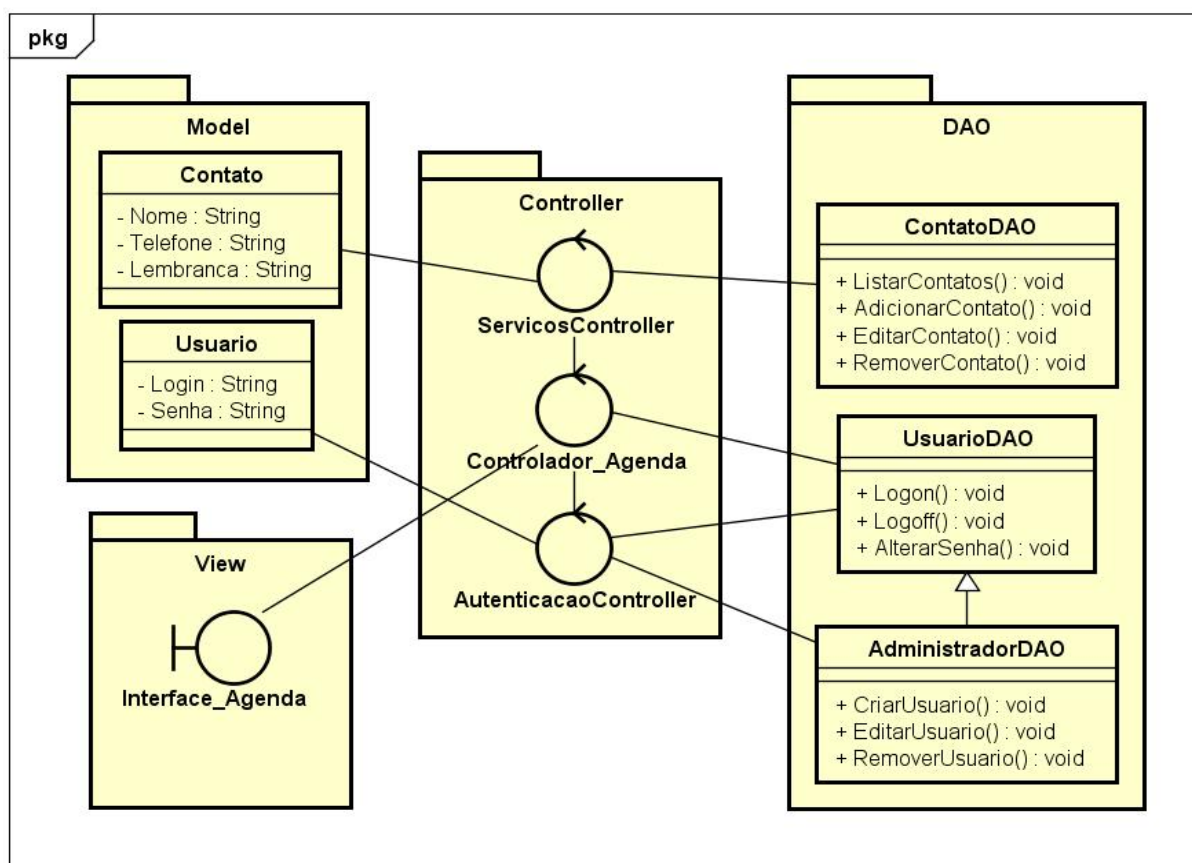


Fonte: Própria autoria.

Na figura 03, observou-se o diagrama das classes do sistema, no qual a divisão das camadas do padrão MVC foi aplicada. A interface do sistema (view) se comunicou com as classes usuário e contato (model) através de controladores (controller). Para realizar a persistência dos dados no banco de dados, foi utilizado o padrão DAO. As classes UsuarioDAO e ContatoDAO (DAO) possuíam os métodos para persistência dos dados em banco de dados, utilizados pelos controladores (controllers).

Através do Diagrama de Pacotes (figura 04) a seguir, foi possível observar a divisão das camadas do Padrão MVC e o Padrão DAO, separados nos pacotes:

Figura 04 – Diagrama de pacote

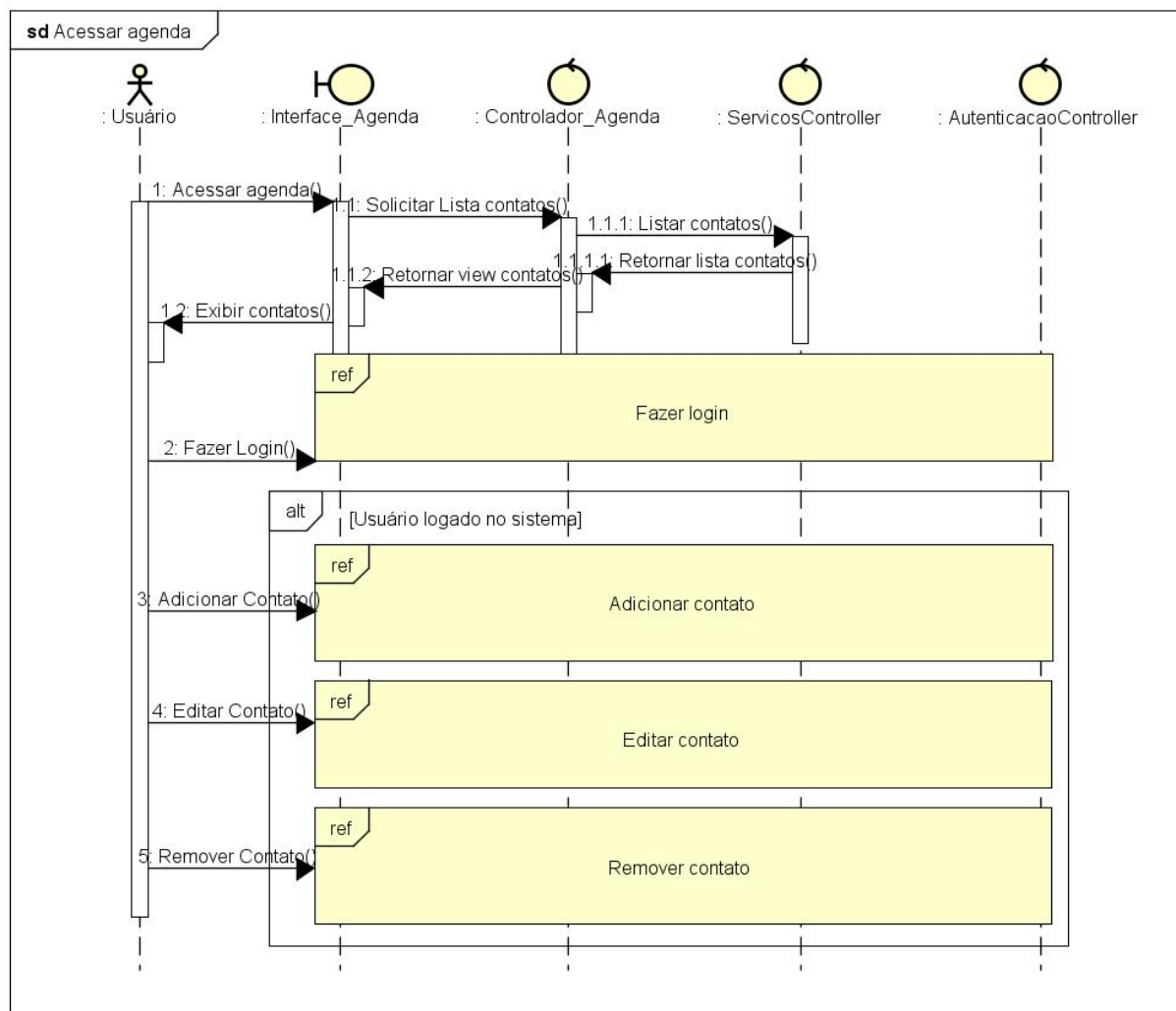


Fonte: Própria autoria.

Na figura 04, foi possível identificar as camadas que compõem o sistema. Neste diagrama observou-se que as classes Contato e Usuario fazem parte do domínio da aplicação, as classes ContatoDAO e UsuarioDAO possuíam os métodos para persistência dos dados no banco de dados, quem eram os controladores do sistema e que a camada View foi composta pela interface\_Agenda.

Nos diagramas de sequência a seguir (figuras 05 e 06), observou-se as etapas percorridas pelo sistema, quando o usuário acessa o sistema, realiza login no sistema, e adiciona um novo contato ao sistema:

Figura 05 – Diagrama de sequência: Acessar agenda



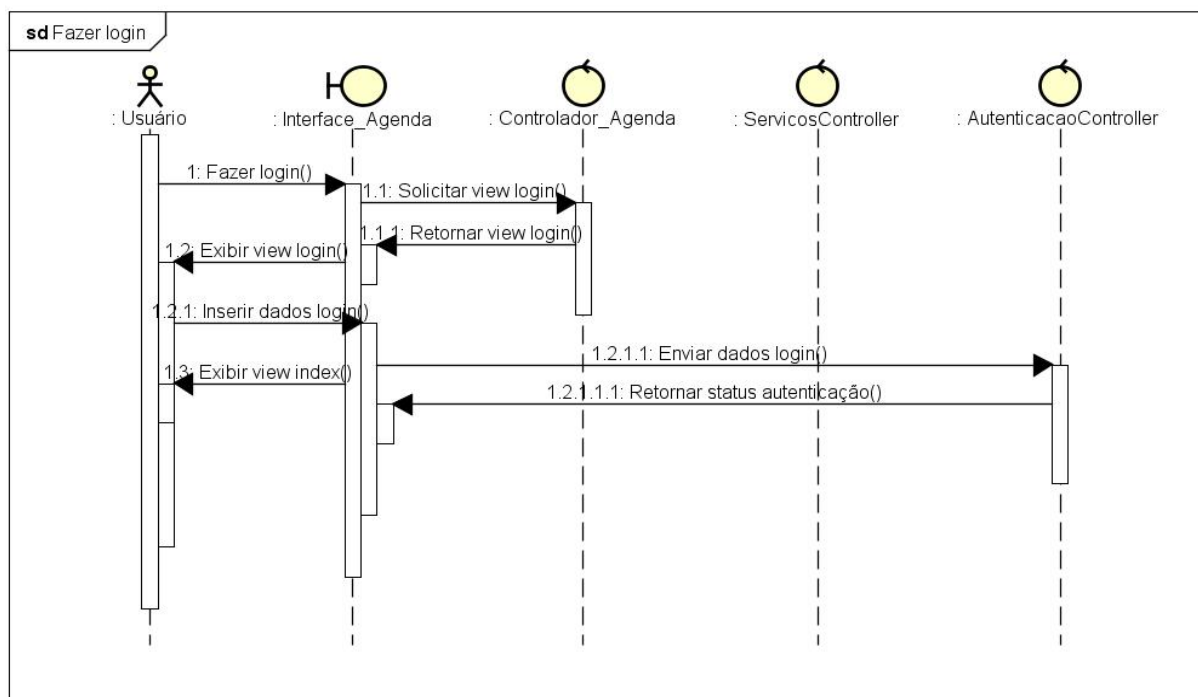
Fonte: Própria autoria.

Conforme pôde-se observar no diagrama de sequência apresentado na figura 05, quando o usuário acessa o sistema, a interface do sistema solicita aos controladores a lista dos contatos salvos no banco de dados, que por sua vez, fez a consulta no banco e retornou a lista de contatos para a interface do sistema, que exibiu a *view* home com a lista de contatos telefônicos ao usuário. Também foi possível observar que na página inicial do sistema, o usuário teve a opção de realizar o login, e, se já estivesse logado, as

opções de adicionar um novo contato, ou editar ou remover os contatos já existentes no sistema, seriam exibidas.

Qualquer usuário, mesmo que não esteja logado no sistema, tem acesso à consulta dos dados de contatos no sistema. Para realização de operações de inclusão, exclusão ou alteração dos dados, é necessário que o usuário esteja logado no sistema, caso contrário, estas opções não são exibidas. Apenas o usuário administrador do sistema pode realizar inclusão, exclusão, consulta e alteração dos usuários do sistema.

Figura 06 – Diagrama de sequência: Fazer login

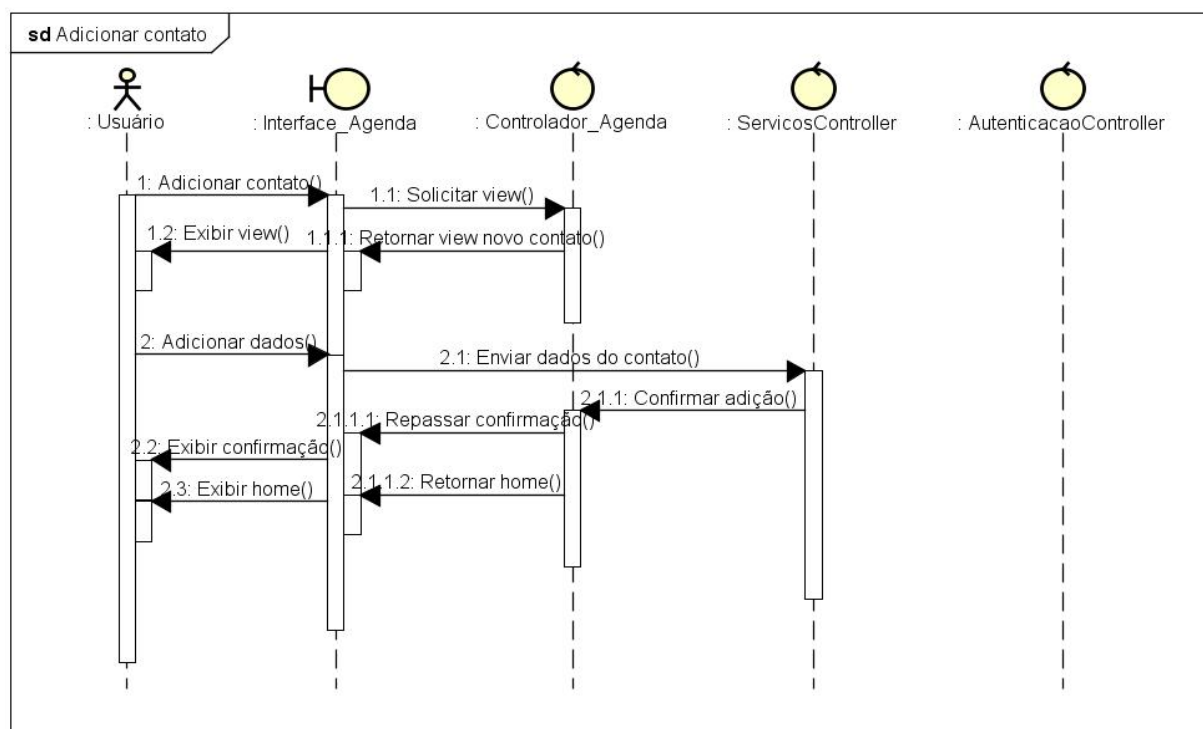


Fonte: Própria autoria.

A figura 06, demonstrou as etapas percorridas pelo sistema para realização do login de usuário. Quando o usuário acessou a opção “fazer login”, a interface do sistema solicitou a *view* com o formulário de *login* ao controlador\_Agenda, que retornou a *view* para a interface, que exibiu-a ao usuário. Em seguida, o usuário inseriu os dados de login no formulário, a *view* enviou os dados do formulário para o controlador de autenticações, que por sua vez, retornou o status da autenticação a *view*, que a exibiu para o usuário.

Para adicionar um novo contato ao banco de dados, o sistema percorreu as etapas descritas na figura 07:

Figura 07 – Diagrama de sequência: Adicionar contato.



Fonte: Própria autoria.

Quando o usuário está logado no sistema, é exibida a opção para adição de um novo contato. Ao acessar esta função, a interface do sistema solicita a *view* ao Controlador\_Agenda, que retorna a *view* com o formulário para adição de um novo contato. Em seguida, o usuário deverá preencher o formulário com os dados do contato que deseja inserir no sistema. A interface envia os dados para o controlador ServicosController, que adiciona o novo contato ao sistema, e retorna a confirmação de adição do contato no banco de dados, para o Controlador\_Agenda, que por sua vez, repassa a confirmação e retorna a *view* home para a interface do sistema, que exibe a confirmação de adição ao usuário e, em seguida, exibe a *view* home.

### 3.5. IMPLEMENTAÇÃO DO SISTEMA

Conforme citado anteriormente, para a implementação do sistema foram utilizadas as linguagens C# e Asp.net no *backend* da aplicação, Javascript e HTML5 no *frontend* da aplicação. Também foram utilizados as ferramentas .NET Framwork e Entity Framework, que auxiliam na estruturação e divisão das camadas, e na comunicação entre



as classes da aplicação. O sistema foi dividido em camadas, conforme a aplicação dos padrões MVC e DAO. Cada camada tem sua responsabilidade no sistema, onde a camada de Visão é responsável pela interface do sistema, cuidando da exibição das telas, a Camada controle é responsável pelo controle das atividades da aplicação, a camada modelo é responsável pelas regras de negócio do sistema, e a camada DAO é responsável pela persistência dos dados no banco de dados.

O sistema foi desenvolvido em dois módulos: O módulo de Usuários realiza os serviços de autenticação e controle de acesso dos usuários ao sistema, e o módulo de Contatos realiza o serviço de gerenciamento dos contatos telefônicos no sistema. Os dois módulos são interligados, de modo que o módulo de Usuários realiza o controle do acesso aos dados referentes ao módulo Contatos.

#### 3.5.1. Aplicação do padrão MVC

MVC é o acrônimo para *Model-View-Controller*, que são justamente as camadas de divisão do sistema, proposta por este padrão. No sistema desenvolvido, dentro da camada *View* estavam as páginas HTML do sistema, que foram responsáveis pela exibição das funcionalidades do sistema, dentro do navegador. Toda interação com o usuário é feita através da camada *view*. Na camada *controller*, estavam os *controllers* da aplicação, que foram os responsáveis por receber e responder as requisições do usuário, receber os dados da *view* e enviá-los para o banco o banco de dados, através da camada DAO, e enviar os dados recebidos do banco de dados, para exibição na *view*. Na camada *Model* estavam contidas as classes Contato e Usuário, que foram responsáveis pela regra de negócio do sistema. Elas definiram quais atributos e métodos cada contato e usuário deverão possuir

#### 3.5.2. Aplicação do padrão DAO

DAO é o acrônimo para *Data Access Object*, que traduzindo para o português seria objeto de acesso a dados, o principal objetivo para aplicação do padrão DAO, é abstrair da camada de negócios, o mecanismo de persistência de dados utilizado na

aplicação. Foi criada uma camada para aplicação desse padrão, e esta camada é responsável pela persistência dos dados no banco de dados MySQL. Nesta camada estão contidas as classes ContatoDAO, UsuarioDAO e AdministradorDAO, que são responsáveis pelas operações de consulta, inserção, alteração e exclusão de dados no banco de dados.

### *3.5.3. Aplicação do padrão Baixo Acoplamento*

O padrão Baixo Acoplamento está relacionado ao quão dependente um elemento é de outros elementos, e com a aplicação deste padrão, os elementos do sistema não dependem de muitos outros. Isso não quer dizer que não haja dependências entre os elementos do sistema, mas que as dependências sejam minimizadas. Durante o desenvolvimento do projeto, um dos princípios que foram seguidos foi o do padrão Baixo Acoplamento, e conforme pode-se observar na figura 03, apresentada anteriormente, os relacionamentos entre as classes foram minimizados, para que se mantivesse o baixo acoplamento entre os elementos do sistema, e assim, tornar o código mais simples de ser compreendido, facilitar sua reutilização, facilitar a manutenção, evitar modificações forçadas devido a alterações feitas em classes relacionadas, entre outras questões trazidas pela utilização deste padrão.

### *3.5.4. Aplicação do padrão Alta Coesão*

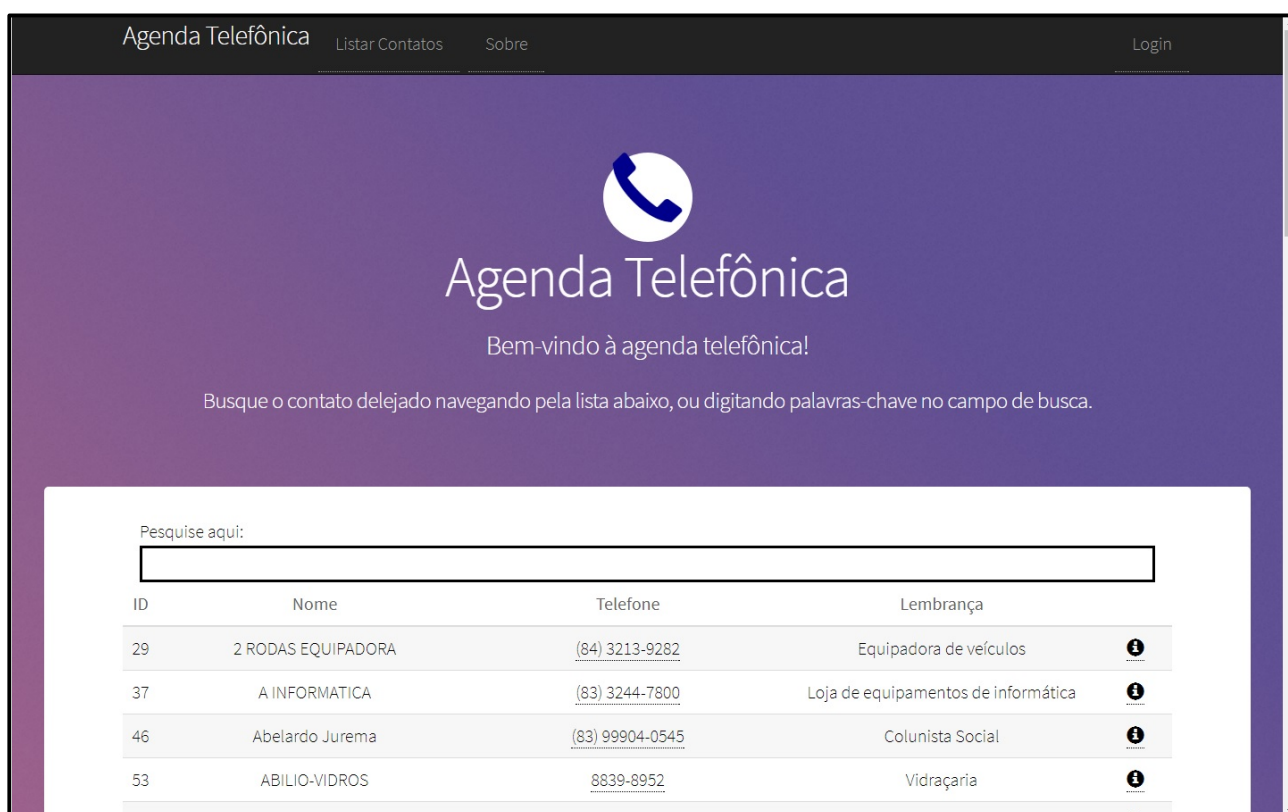
O padrão Alta Coesão diz respeito ao quão relacionadas e focalizadas estão as responsabilidades de um elemento. Quando um elemento possui responsabilidades que estão relacionadas entre si, e ele não executa um grande volume de trabalho, então este tem coesão alta. O padrão Alta Coesão está ligado ao princípio da responsabilidade única, que diz que uma classe deve ter apenas uma responsabilidade, não assumindo responsabilidades de outras classes. Assim como o padrão Baixo Acoplamento, durante o desenvolvimento do projeto, também foram seguidos os princípios do padrão Alta Coesão, e conforme observou-se na figura 03, apresentada anteriormente, foram atribuídas responsabilidades aos elementos do sistema de maneira coesa, sem que haja sobrecarga de responsabilidades aos elementos.

### 3.6. INTEGRAÇÃO DE SISTEMA

Conforme apresentado anteriormente, o sistema é composto por dois módulos: O módulo de Usuários realiza os serviços de autenticação e controle de acesso dos usuários ao sistema, e o módulo de Contatos realiza o serviço de gerenciamento dos contatos telefônicos no sistema. Os dois módulos se integram, onde o módulo de Usuários faz o controle de acesso aos dados do módulo de Contatos.

A figura 08 exibe a tela inicial do sistema. Nesta tela são exibidas as opções de exibir a lista de contatos cadastrados no sistema, exibindo os campos ID, nome, telefone e lembrança do contato, os dados completos de cada contato cadastrado, individualmente, e a página que fala sobre o sistema.

Figura 08 – Tela inicial do sistema.



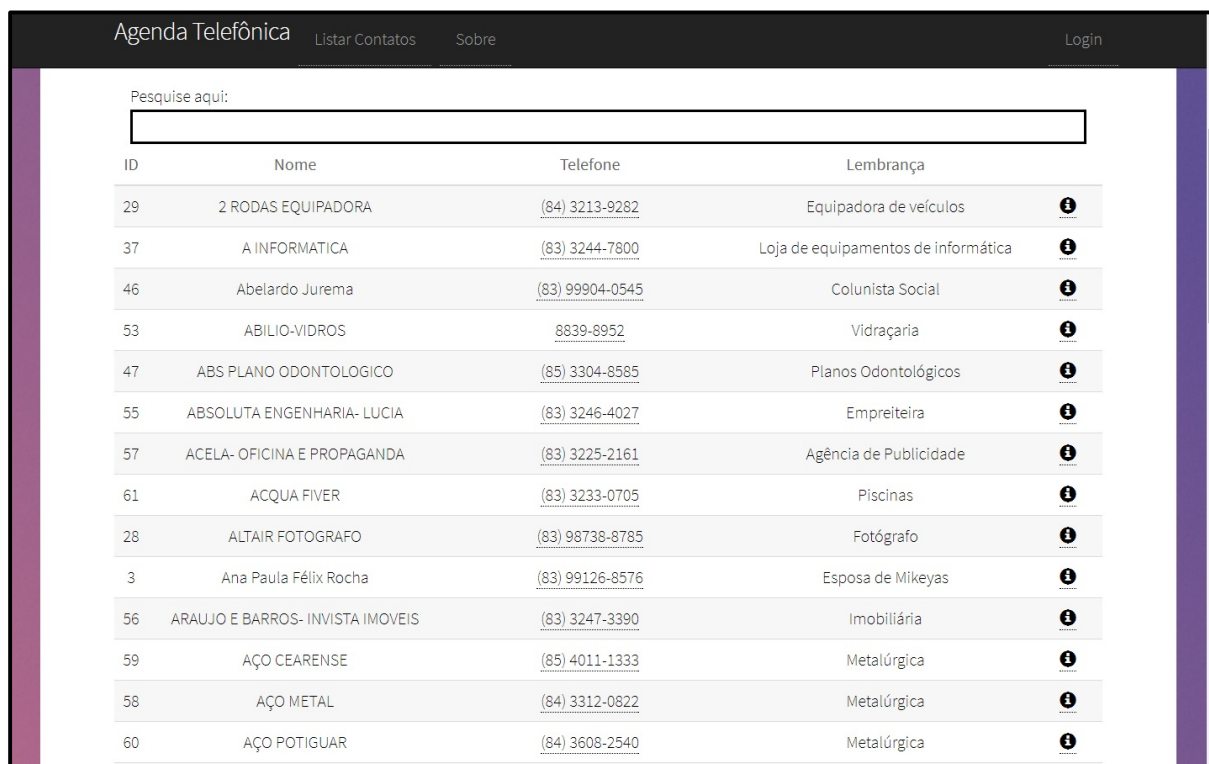
*Fonte: Própria autoria.*

Ao acessar o sistema, sem que tenha realizado o login, o usuário apenas visualizou a lista de contatos cadastrados no sistema e tem acesso aos dados cadastrados de cada contato individualmente, e para ter acesso a outras opções, é necessário que o usuário realize login no sistema. O sistema possui uma Navbar para auxiliar ao usuário durante a navegação, e é através dela que o usuário pode acessar as

opções de listar os contatos cadastrados no sistema, a página que fala sobre o sistema, e fazer o login no sistema.

A figura 09 exibe a tela com a lista de contatos cadastrados no sistema. Na lista foram exibidos o ID, nome, telefone, lembrança e o botão para exibir o cadastro completo de cada contato cadastrado no sistema. As opções para edição e exclusão dos contatos da lista apenas são exibidas quando o usuário realizar o login no sistema.

Figura 09 – Lista de contatos cadastrados no sistema.



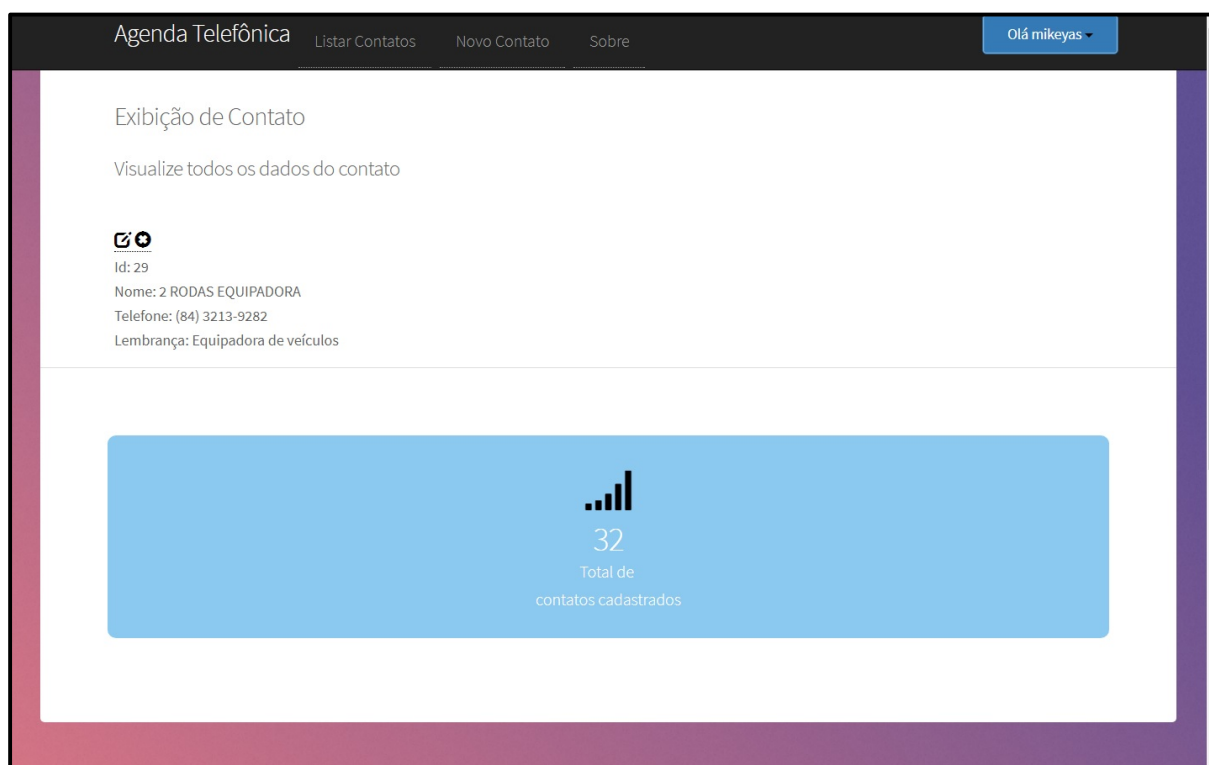
ID	Nome	Telefone	Lembrança	
29	2 RODAS EQUIPADORA	(84) 3213-9282	Equipadora de veículos	
37	A INFORMATICA	(83) 3244-7800	Loja de equipamentos de informática	
46	Abelardo Jurema	(83) 99904-0545	Colunista Social	
53	ABILIO-VIDROS	8839-8952	Vidraçaria	
47	ABS PLANO ODONTOLOGICO	(85) 3304-8585	Planos Odontológicos	
55	ABSOLUTA ENGENHARIA- LUCIA	(83) 3246-4027	Empreiteira	
57	ACELA- OFICINA E PROPAGANDA	(83) 3225-2161	Agência de Publicidade	
61	ACQUA FIVER	(83) 3233-0705	Piscinas	
28	ALTAIR FOTOGRAFO	(83) 98738-8785	Fotógrafo	
3	Ana Paula Félix Rocha	(83) 99126-8576	Esposa de Mikeyas	
56	ARAUJO E BARROS- INVISTA IMOVEIS	(83) 3247-3390	Imobiliária	
59	AÇO CEARENSE	(85) 4011-1333	Metalúrgica	
58	AÇO METAL	(84) 3312-0822	Metalúrgica	
60	AÇO POTIGUAR	(84) 3608-2540	Metalúrgica	

Fonte: Própria autoria.

Ao clicar no botão para exibir o cadastro completo do contato cadastrado no sistema, o sistema direciona o usuário para a tela que apresenta os detalhes do contato escolhido pelo usuário, conforme apresentado na figura 10, exibida a seguir:

Na figura 10 é exibida a tela de detalhes do contato cadastrado no sistema, nela são exibidos o detalhamento dos dados cadastrados para cada usuário cadastrado no sistema. Caso o usuário não esteja autenticado no sistema, essa tela apenas exibirá os dados, e após a autenticação do usuário, o sistema disponibilizará as opções de edição e exclusão do contato, contidas nesta tela.

Figura 10 – Tela de detalhes do contato cadastrado no sistema.



*Fonte: Própria autoria.*

Para realizar login no sistema, o usuário deve acessar a opção “login” contida na Navbar do site. Ao acessar esta opção, o usuário será direcionado para a tela de login, exibida na figura 11.

Ao acessar a tela de login, apresentada na figura 11, o usuário terá acesso ao formulário de login, e precisará inserir um usuário e senha válidos para poder se autenticar no sistema. Ao receber o login e senha do usuário, o sistema realizará um processo de validação desses dados, para verificar se os mesmos são válidos, e para a senha, o sistema ainda utiliza um algoritmo MD5 para realizar a criptografia da senha recebida, e compará-la com a senha criptografada contida no banco de dados. Após o processo de validação dos dados, caso estes dados não estejam contidos no banco de dados do sistema, o sistema retornará uma mensagem informando que o usuário ou senha estão incorretos, e caso o usuário e a senha informados estejam corretos, o sistema fará a autenticação do usuário no sistema, e redirecionará o sistema para sua tela inicial.

Figura 11 – Tela de login no sistema.

A interface de login do sistema 'Agenda Telefônica' apresenta uma barra superior com o nome do sistema e links para 'Listar Contatos' e 'Sobre'. O formulário principal, intitulado 'Login do sistema', solicita o nome de usuário e senha. Ele contém campos de entrada para 'Digite seu login:' e 'Digite sua senha:', seguidos por um botão 'Enviar'. Abaixo do formulário, uma barra azul exibe um ícone de gráfico de barras e o texto '32 Total de contatos cadastrados'.

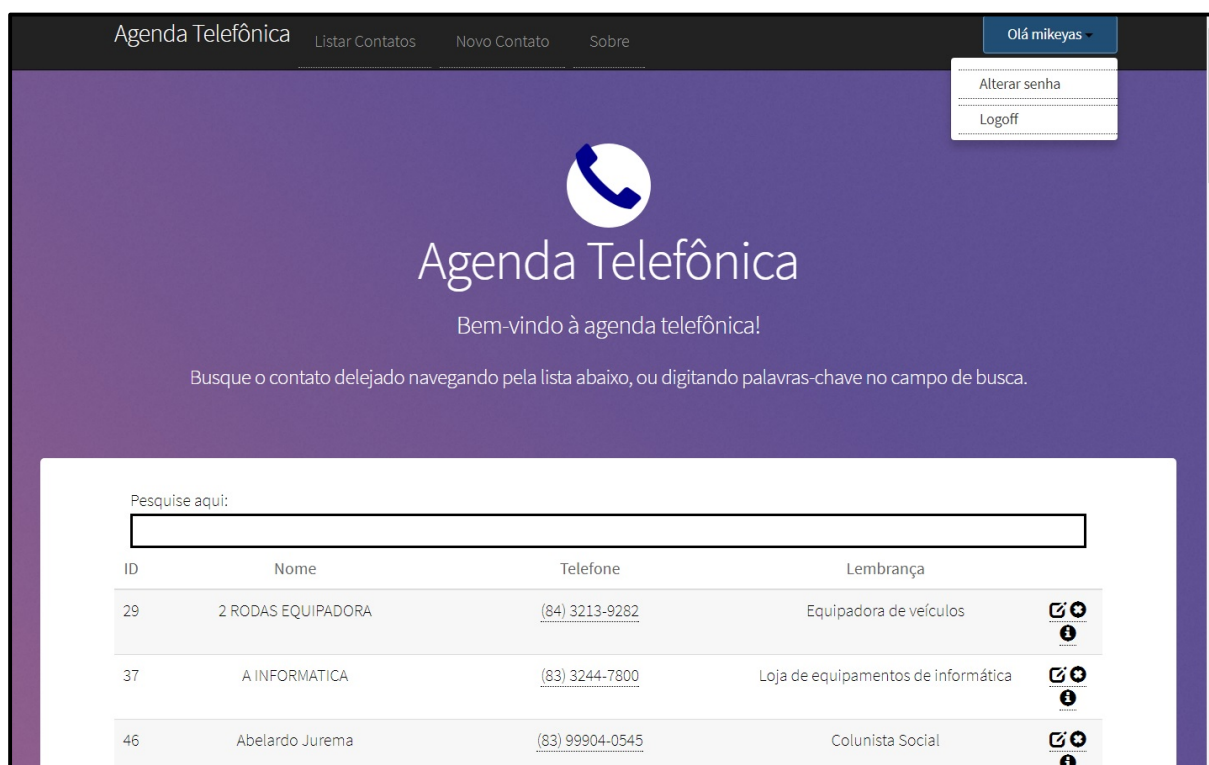
*Fonte: Própria autoria.*

Quando o usuário está autenticado no sistema, são exibidas outras funcionalidades do sistema, conforme demonstrado na figura 12, na página a seguir. Para usuários autenticados, que não sejam o administrador do sistema, além das opções já exibidas sem que o usuário esteja autenticado, o sistema exibirá as opções para adicionar um novo contato, editar ou excluir um contato já existente no sistema, e também as opções para alterar a senha do usuário e realizar *logout* no sistema.

Caso o usuário está autenticado no sistema seja o administrador, além das opções exibidas para o usuário simples, o administrador também terá acesso à página de gerência de usuários do sistema, conforme apresentado na figura 13, na página a seguir.

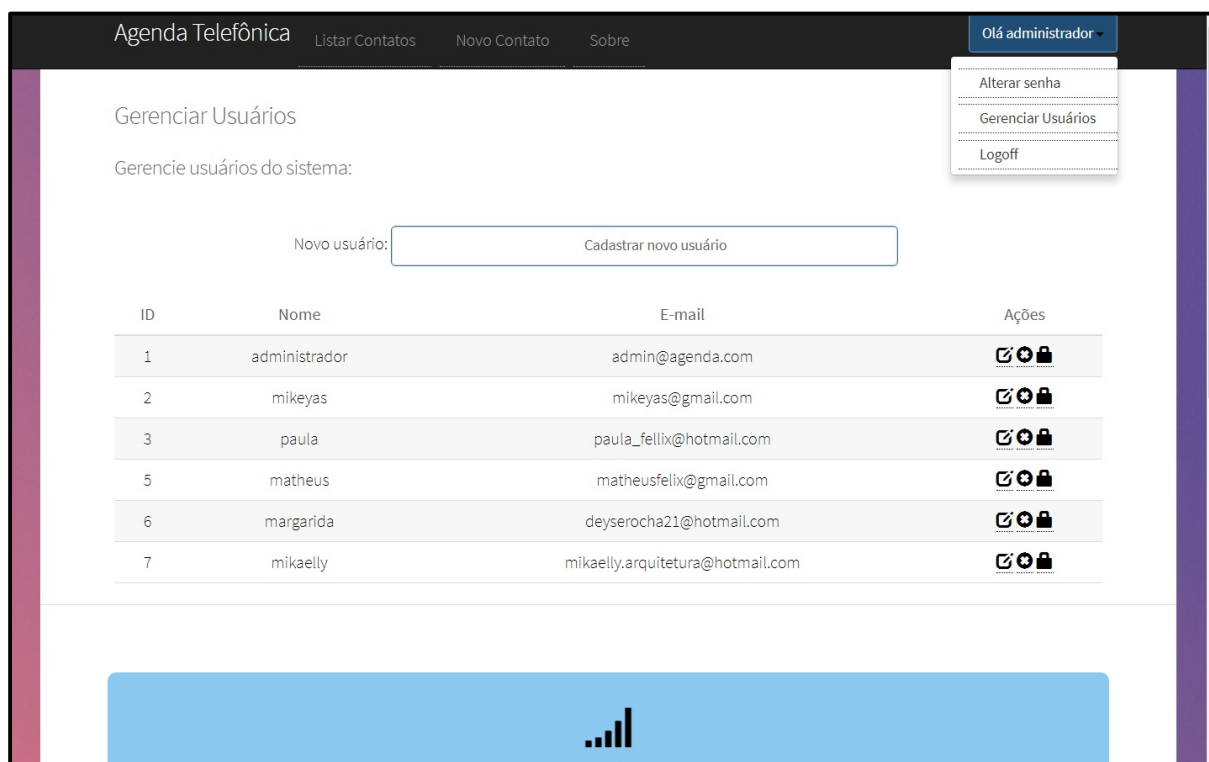
Na figura 13, são disponibilizadas ao administrador do sistema as opções de cadastrar um novo usuário ao sistema, editar, excluir ou resetar a senha de um usuário cadastrado no sistema. Estas opções são disponibilizadas apenas para o usuário administrador do sistema. O sistema foi desenvolvido com apenas um administrador do sistema, e não é possível adicionar outros usuários como administradores do sistema, nem alterar ou excluir o usuário administrador, sendo permitido apenas alterar a senha do administrador.

Figura 12 – Tela inicial do sistema para usuários autenticados.



Fonte: Própria autoria.

Figura 13 – Tela gerência de usuários do sistema.



Fonte: Própria autoria.

Na figura 14 é exibida a tela para adição de um novo contato ao sistema. Para adicionar um novo contato ao sistema, o usuário precisa preencher os campos com o nome, telefone e lembrança para o contato desejado. Todos os três campos são de preenchimento obrigatório.

Figura 14 – Tela adicionar novo contato ao sistema.

A interface de usuário para adicionar um novo contato. No topo, há uma barra de navegação com o título "Agenda Telefônica" e links para "Listar Contatos", "Novo Contato" e "Sobre". Um botão "Olá mikeyas" está no canto superior direito. O formulário principal contém o título "Cadastrar novo Contato.", a instrução "Adicione um novo contato à agenda:", e três campos de entrada rotulados "Nome do Contato:", "Telefone do Contato:" e "Lembrança do contato:". Abaixo dos campos está um botão "Salvar Contato". Na base da tela, há uma barra azul com um ícone de sinal de celular.

*Fonte: Própria autoria.*

Na figura 15 é exibida a tela para edição de um contato já existente no sistema. Ao clicar no botão “Editar contato” o sistema exibirá a página para edição do contato, com os campos nome, telefone e lembrança do contato já preenchidos com os dados do contato salvos no banco de dados. O usuário deverá alterar as informações desejadas e clicar no botão “Editar contato”. Lembrando que todos os campos são de preenchimento obrigatório.



Figura 15 – Tela editar contato do sistema.

Agenda Telefônica

Listar Contatos Novo Contato Sobre

Olá mikeyas

### Edição de Contato

Edite um contato da agenda.

2 RODAS EQUIPADORA

(84) 3213-9282

Equipadora de veículos

Editar Contato

32  
Total de

Fonte: Própria autoria.

Figura 16 – Remover contato do sistema.

Agenda Telefônica

Listar Contatos

10.20.30.103:49678 diz:  
Deseja realmente excluir contato?

OK Cancelar

Olá mikeyas

Pesquise aqui:

ID	Nome	Telefone	Lembrança	
29	2 RODAS EQUIPADORA	(84) 3213-9282	Equipadora de veículos	
37	A INFORMATICA	(83) 3244-7800	Loja de equipamentos de informática	
46	Abelardo Jurema	(83) 99904-0545	Colunista Social	
53	ABILIO-VIDROS	8839-8952	Vidraçaria	
47	ABS PLANO ODONTOLOGICO	(85) 3304-8585	Planos Odontológicos	
55	ABSOLUTA ENGENHARIA- LUCIA	(83) 3246-4027	Empreiteira	

Fonte: Própria autoria.

Na figura 16 é exibida a funcionalidade de remoção de um contato do sistema. Quando o usuário clicar no botão “Remover contato”, o sistema exibirá a pergunta: “deseja realmente remover o contato do sistema?”, com as opções “OK” e “Cancelar”, e o usuário deverá escolher “OK” para prosseguir com a remoção, ou “Cancelar” para cancelar a ação.

Para alterar a senha, o usuário acessa a opção “alterar senha”, sendo direcionado para a página de alterar a senha, conforme exibido na figura 17. Para realizar esta ação, o usuário precisa informar a senha atual, informar a nova senha e repetir a nova senha, depois clicar no botão enviar. O sistema verifica se a senha atual está correta, depois verifica se as duas novas senhas coincidem, caso os dados estejam corretos, a senha é alterada e o sistema retorna uma confirmação de alteração.

Figura 17 – Altera senha do usuário.

A imagem mostra a interface de usuário para a alteração de senha. No topo, há uma barra de navegação com o título "Agenda Telefônica" e links para "Listar Contatos", "Novo Contato" e "Sobre". No canto superior direito, há um botão "Olá administrador". O conteúdo principal da página é o formulário "Cadastro de Usuários", com o subtítulo "Adicione usuários ao sistema:". O formulário contém quatro campos de entrada: "Digite o Usuário:", "Digite a Senha:", "Repita a Senha:" e "Digite o E-mail:". Abaixo dos campos, há um botão "Enviar". No rodapé da interface, há uma barra azul com um ícone de sinal de celular.

Fonte: Própria autoria.

Apenas o usuário administrador tem permissões para realizar operações de inclusão, exclusão, edição e *reset* de senha de usuários. Na figura 18 é exibida a página para cadastramento de um novo usuário no sistema. Para cadastrar um novo usuário no sistema, o administrador precisa informar o nome, a senha, repetindo-a no campo seguinte e o e-mail do novo usuário. O sistema verifica se o nome de usuário ou e-mail

informado já existem, e se as senhas digitadas coincidem, e caso a validação dos dados tenha sucesso, o sistema adicionará o novo usuário ao sistema, retornando a confirmação da adição do contato ao administrador.

Figura 18 – Cadastrar um novo usuário do sistema.

The screenshot shows a web application titled 'Agenda Telefônica' with a navigation bar containing 'Listar Contatos', 'Novo Contato', and 'Sobre'. A user is logged in as 'Olá administrador'. The main content area is titled 'Cadastro de Usuários' and contains the instruction 'Adicione usuários ao sistema:'. Below this, there are four input fields labeled 'Digite o Usuário:', 'Digite a Senha:', 'Repita a Senha:', and 'Digite o E-mail:'. At the bottom of the form is an 'Enviar' button.

Fonte: Própria autoria.

Figura 19 – Editar um usuário existente.

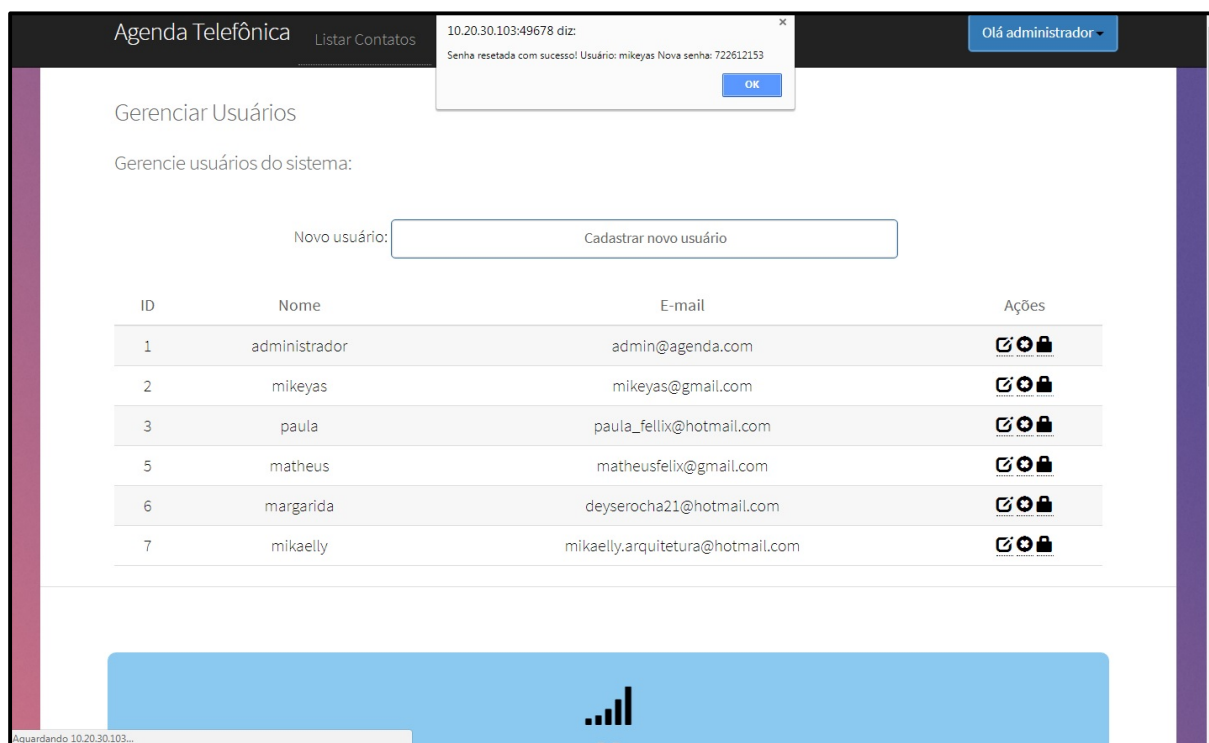
The screenshot shows the same 'Agenda Telefônica' web application, but the main content area is titled 'Cadastro de Usuários' with the instruction 'Edite o usuário do sistema:'. The form has two input fields: 'Digite o Usuário:' with the value 'mikeyas' and 'Digite o E-mail:' with the value 'mikeyas@gmail.com'. An 'Enviar' button is at the bottom. Below the form, there is a blue box containing a bar chart icon, the number '32', and the text 'Total de contatos cadastrados'.

Fonte: Própria autoria.

Para editar um usuário, o administrador precisa clicar no botão 'editar usuário', na tela gerência de usuários do sistema (figura 13). A figura 19 mostra a tela para editar o usuário. Apenas o nome de usuário ou e-mail podem ser editados, e o sistema realiza a validação dos dados antes de alterá-los.

Para remover um usuário, o administrador precisa clicar no botão 'remover usuário', na tela gerência de usuários do sistema (figura 13). Ao solicitar a remoção do usuário, o sistema verificará se o usuário que está sendo removido não é o administrador, e caso não seja, ele removerá o usuário do sistema.

Figura 20 – Resetar senha de um usuário.



Fonte: Própria autoria.

Para resetar a senha de um usuário no sistema, o administrador precisa clicar no botão 'resetar senha', na tela gerência de usuários do sistema (figura 20). Quando o administrador solicitar o *reset* da senha de um usuário listado, o sistema verificará se o *reset* de senha é do administrador, e caso não seja, ele executará o *reset* da senha, retornando a mensagem de confirmação do *reset* de senha, e informar o nome de usuário e a nova senha, conforme observou-se na figura 20.

### 3.7. OPERAÇÃO E MANUTENÇÃO

Geralmente esta etapa é a mais demorada no processo de desenvolvimento de *software* em cascata, pois esta é a etapa em que o sistema é instalado e colocado em uso. Sua manutenção, segundo Sommerville (2011, p. 21), é a identificação e correção de *bugs* que não foram identificados durante as etapas do projeto, bem como realizar melhorias nas unidades do sistema e implantação de novos requisitos descobertos.

Partindo deste princípio, o processo de implantação do sistema de gerenciamento de contatos telefônicos foi iniciado na empresa Planc Engenharia e Incorporações Ltda, e está sendo utilizado por alguns dos colaboradores da sede da organização, no intuito de contribuir para a implantação do mesmo. O banco de dados está sendo populado com os dados contidos no sistema antigo, os usuários estão sendo criados, e estão sendo realizados treinamentos com os colaboradores designados pela empresa. O sistema ainda está em processo de configuração e ajustes necessários para o uso, e estima-se que até o final de dezembro de 2017, o sistema esteja em pleno funcionamento, com todas as configurações necessárias realizadas, onde, a partir de janeiro de 2018, seu uso será expandido para todos os colaboradores. O mesmo funciona apenas na rede interna da empresa, e não possui forma de acesso externo. A manutenção do sistema será feita de acordo com a necessidade da organização, e conforme os níveis de prioridade/urgência estipulados, caso venham surgir defeitos no sistema, ou necessidade de realização de melhorias nas unidades do sistema, ou ainda com a implantação de novos requisitos que surjam durante a utilização do sistema.

#### 4. CONCLUSÃO

Os principais resultados obtidos com a pesquisa envolveram, primeiro, a importância do planejamento do projeto e utilização de metodologias que auxiliam o planejamento e desenvolvimento do projeto, tais como levantamento de requisitos, modelagem UML, Padrões de Projetos, modelos de processos de software, etc. Segundo, a utilização de diagramas UML para facilitar o entendimento dos requisitos e do funcionamento do sistema, como um todo, e direcionar o desenvolvimento do projeto. Terceiro, a aplicação dos Padrões de Projetos como técnica de padronização que pode ser adotada para facilitar a padronização dos projetos que envolvam a construção de sistemas *web* que utilizam o paradigma da Orientação a Objetos. Quarto, possibilitou perceber a viabilidade em se construir aplicações com linguagem e banco de dados gratuitos. Quinto, permitiu verificar que com a mesma estrutura de equipamentos (hardware), é possível melhorar substancialmente a eficiência do sistema de gerenciamento de contatos telefônicos, podendo facilitar o dia a dia dos colaboradores, aliviando a carga de trabalho e agilizando a consulta às informações. Finalmente, apontou soluções extensíveis a outras ferramentas utilizadas na organização.

Com a implantação do sistema de gerenciamento de contatos telefônicos, desenvolvido no decorrer deste projeto, a organização obteve um sistema simplificado, de fácil acesso por qualquer colaborador com poucas noções de informática, ágil e com interface amigável. Facilmente adaptável a futuras necessidades e de simples entendimento por outros desenvolvedores, devido as metodologias utilizadas no desenvolvimento do projeto. Futuramente, o trabalho desenvolvido tem potencial para transformar-se em uma ferramenta comercial, pois com poucas modificações ele pode facilmente ser adaptado as mais diversas aplicações em um sistema web para gerenciamento de contatos telefônicos, ou ainda, poderá ser ampliado, adicionando novos módulos, e compor a intranet da organização em que foi implantado.

Para trabalhos futuros, poderão ser implementados: sistema de controle de bens imobilizados da organização, utilizando-se como base o sistema atual, e realizando alterações em seus requisitos, para que o mesmo atenda tal demanda, visto que a organização possui um grande volume de bens imobilizados em estoque, ou ainda um sistema dinâmico de controle das tabelas de preços dos empreendimentos da organização, pois atualmente a mesma faz uso de tabelas confeccionadas em planilhas

eletrônicas, e o sistema atual poderia atender tal finalidade sem grandes alterações em sua estrutura.

## REFERÊNCIAS

FREEMAN, Eric; FREEMAN, Elisabeth. **Use a Cabeça! Padrões de Projetos**. 2.ed. Rio de Janeiro: Alta Books, 2009.

GAMMA, Erich; HELMET, Richard; JOHNSON, Ralph; VLISSIDES, John. **Padrões de projeto**: soluções reutilizáveis de software orientado a objetos. Porto Alegre: Bookman, 2005.

GUEDES, Gilleanes T. A. **UML 2**: uma abordagem prática. São Paulo: Novatec Editora, 2011.

HORSTMANN, Cay. **Padrões e projetos orientados a objetos**. 2.ed. Porto Alegre: Bookman, 2007.

LARMAN, Craig. **Utilizando UML e Padrões**: uma introdução à análise e ao projeto orientados a objetos e ao desenvolvimento iterativo. Porto Alegre: Bookman, 2007.

LIBERTY, Jesse; HOROVITZ, Alex. **Programando .NET 3.5**. Rio de Janeiro: Alta Books, 2009.

LIMA, Edwin. **C# e .Net para desenvolvedores**. Rio de Janeiro: Campus, 2002.

MICROSOFT. **Introdução à linguagem C# e o .NET Framework**. Disponível em: <[https://msdn.microsoft.com/pt-br/library/z1zx9t92\(v=vs.110\).aspx](https://msdn.microsoft.com/pt-br/library/z1zx9t92(v=vs.110).aspx)>. Acessado em 11/06/2017, às 15 h.

MILANI, André. **MySQL**: Guia do programador. São Paulo: Novatec Editora, 2006.

MYSQL. **Manual de Referência do MySQL 4.1**. Disponível em: <[dev.mysql.com](http://dev.mysql.com)>. Acessado em 11/06/2017, às 15 h.

SARDAGNA, Marcelo; VAHLICK, Adilson. **Aplicação do padrão Data Access Object (DAO) em projetos desenvolvidos com DELPHI**. Blumenau: Universidade Regional de Blumenau, 2007.

SILVA, Maurício Samy. **Criando Sites com HTML**: sites de alta qualidade com HTML e CSS. São Paulo: Novatec Editora, 2008.

SILVA, Maurício Samy. **HTML 5**: A linguagem de marcação que revolucionou a web. 2.ed. São Paulo: Novatec Editora, 2014.

SILVA, Maurício Samy. **Javascript: Guia do programador**. São Paulo: Novatec Editora, 2010.

SOMMERVILLE, Ian. **Engenharia de Software**. 9.ed. São Paulo: Pearson, 2011.

TRINDADE, Joana M. F.; FISCHER, Leonardo Garcia. **Estudo e Aplicação de Padrões**. Porto Alegre: Universidade Federal do Rio Grande do Sul, 2007.