**UNIVERSIDADE PAULISTA**

**ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

**EMERSON GASPARINI JUNIOR - N19044-2**

**HEYTTOR FLEMMING ANTUNES SILVA – D3402F-2**

**IGOR MENDES MEDEIROS – N117DB-2**

**JULLIANN ANDRÉ RAMOS – N139JF-2**

**MURILO DE ALMEIDA LIMA – D223JC-6**

**SISTEMA DE HELP-DESK**

**EVOLUTION**

**SOROCABA-SP**

**2018**

**EMERSON GASPARINI JUNIOR - N19044-2**

**HEYTTOR FLEMMING ANTUNES SILVA – D3402F-2**

**IGOR MENDES MEDEIROS – N117DB-2**

**JULLIANN ANDRÉ RAMOS – N139JF-2**

**MURILO DE ALMEIDA LIMA – D223JC-6**

**SISTEMA DE HELP-DESK**

**Trabalho de conclusão do semestre na forma de projeto integrado multidisciplinar (PIM) apresentado à Universidade Paulista - UNIP**

**Orientador: Todos os Professores do Semestre**

**SOROCABA-SP**

**2018**

**EMERSON GASPARINI JUNIOR - N19044-2**

**HEYTTOR FLEMMING ANTUNES SILVA - D3402F-2**

**IGOR MENDES MEDEIROS - N117DB-2**

**JULLIANN ANDRÉ RAMOS - N139JF-2**

**MURILO DE ALMEIDA LIMA - D223JC-6**

**UNIP-MAP**

**APLICATIVO DE LOCALIZAÇÃO**

**Trabalho de conclusão do semestre na forma de projeto integrado multidisciplinar (PIM) apresentado à Universidade Paulista - UNIP**

**Orientador: Todos os Professores do Semestre**

**Nota:\_\_\_\_\_\_\_\_\_**

**Richardson Luz**

**Universidade Paulista**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_/\_\_\_**

**Waldir Silva**

**Universidade Paulista**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_/\_\_\_**

**Elizeu Elieber**

**Universidade Paulista**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_/\_\_\_**

**Reverdan Sparinger**

**Universidade Paulista**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_/\_\_\_**

**RESUMO**

Este projeto apresenta o sistema de help-desk Evolution, que foi solicitado à empresa M.I.J.H Systems. O sistema será disponibilizado para uma empresa que prestara serviços de suporte técnico para outras empresas através do sistema de help-desk Evolution. Este sistema foi desenvolvido usando as linguagens C# e ASP.NET, com um banco de dados em SQLServer, que servirá para empresas que contratarem o serviço de suporte técnico terem suporte sobre os produtos da empresa.

Este documento contém toda a parte desenvolvimento do sistema, começando pela análise de requisitos, recursos e custos, fluxogramas e os layouts de tela, passando pelos códigos e banco de dados até os testes feitos no software.

Palavras-chave: Evolution,Help-Desk.

**ABSTRACT**

This paper presents the help – desk system Evolution. The M.I.J.H Systems was requested to develop the Evolution system, which will be available for companies that provide technical support services to others companies by the help – desk system.

The help – desk system Evolution was developed using the C# and ASP.NET languages, with a database SQLServer, that will be useful for companies that hire the technical support service for their products. All the development part of the system is conteined in this paper, beginning from the requirement analysis, resources and costs, flowcharts and the screen layouts, passing through the codes and databases until the tests done in the software.

Key-words: Evolution, Help-Desk.

**INDICE DE ILUSTRAÇÕES**

Figura 1-Tabela de custos 13

Figura 2-Tabela de salario 13

Figura 3-Tabela de linha de base 13

Figura 4-Casos de uso 14

Figura 5-Cronograma 14

Figura 6-Fluxograma do modulo web 15

Figura 7-Fluxograma do modulo desktop 16

Figura 8-Diagrama de atividade do sistema 17

Figura 9-Diagrama de classes 18

Figura 10-Diagrama de sequência do desktop 19

Figura 11-Tela de login do modulo desktop 20

Figura 12-Tela home do desktop 20

Figura 13-Tela de cadastrar usuário 21

Figura 14-Tela de editar usuário 21

Figura 15-Tela de editar cliente 22

Figura 16-Tela de cadastrar funcionário 23

Figura 17-Tela de editar funcionário 23

Figura 18-Tela de cadastrar produto 24

Figura 19-Tela de editar produto 24

Figura 20-Tela de cadastrar e visualizar atendimento 25

Figura 21-Diagrama de sequência do modulo web 26

Figura 22-Tela do login do modulo web 26

Figura 23-Tela do cliente na web 27

Figura 24-Tela do funcionário na web 27

Figura 25-Modelo conceitual 29

Figura 26-Modelo logico 29

Figura 27-APP.Config 30

Figura 28-WEB.Config 31

Figura 29-Tabela de teste 35

Figura 30-Tabela de teste 2 36

Figura 31-Tabela de teste 3 37

Figura 32-Tabela de teste 4 38

Figura 33-Tela de teste 5 39

Figura 34-Tabela teste web 42

Figura 35-Tabela teste web 2 43

Figura 36-Tabela teste web 3 44

Figura 37-Tabela teste web 4 45

Figura 38-Tabela de Atualizações futuras 48

**SUMARIO**

[1.0 Introdução 11](#_Toc530583114)

[2.0 Projeto help desk Evolution 12](#_Toc530583115)

[2.1 Objetivo 12](#_Toc530583116)

[2.2 Suposições e Restrições 12](#_Toc530583117)

[2.3 Recursos 13](#_Toc530583118)

[2.4 custos 13](#_Toc530583119)

[2.5 Datas das Fases e Interações das Linhas Base 13](#_Toc530583120)

[2.6 casos de uso 14](#_Toc530583121)

[2.7 Cronograma 14](#_Toc530583122)

[3.0 Sistema help desk Evolution 15](#_Toc530583123)

[3.0.1 Fluxogramas 15](#_Toc530583124)

[3.0.2 Diagrama de atividades 17](#_Toc530583125)

[3.0.3 Diagrama de classes 18](#_Toc530583126)

[3.1 Desktop 18](#_Toc530583127)

[3.1.1 Diagrama de sequência 19](#_Toc530583128)

[3.1.2 Layout de tela 20](#_Toc530583129)

[3.2 Web 25](#_Toc530583130)

[3.2.1 Diagrama de sequência 26](#_Toc530583131)

[3.2.2 Layouts de telas 26](#_Toc530583132)

[4.0 Desenvolvimento do projeto 28](#_Toc530583133)

[4.1 Código da parte desktop 28](#_Toc530583134)

[4.2 Código da parte web 28](#_Toc530583135)

[5.0 Banco de dados 29](#_Toc530583136)

[5.1 Modelo Conceitual 29](#_Toc530583137)

[5.2 Modelo Logico 29](#_Toc530583138)

[5.3 Modelo Físico 29](#_Toc530583139)

[5.4 Conexões com o banco de dados 30](#_Toc530583140)

[5.4.1 Conexão desktop 30](#_Toc530583141)

[5.4.2 Conexão Web 31](#_Toc530583142)

[6.0 Requisitos de uso e desenvolvimento 32](#_Toc530583143)

[7.0 Parametros de Qualidade 33](#_Toc530583144)

[7.1 Funcionalidades 33](#_Toc530583145)

[7.1.2 Adequação 33](#_Toc530583146)

[7.1.3 Acurácia 33](#_Toc530583147)

[7.1.4 Interoperabilidade 33](#_Toc530583148)

[7.1.5 Segurança 33](#_Toc530583149)

[7.2 Confiabilidade 33](#_Toc530583150)

[7.2.1 Maturidade 33](#_Toc530583151)

[7.2.2 Tolerância a falhas 33](#_Toc530583152)

[7.2.3 Recuperabilidade 33](#_Toc530583153)

[7.3 Usabilidade 34](#_Toc530583154)

[7.3.1 Inteligibilidade 34](#_Toc530583155)

[7.3.2 Apreensibilidade 34](#_Toc530583156)

[7.4 Manutenibilidade 34](#_Toc530583157)

[7.4.1 Analisabilidade 34](#_Toc530583158)

[7.4.2 Modificabilidade 34](#_Toc530583159)

[7.4.3 Estabilidade 34](#_Toc530583160)

[8.0 Testes 35](#_Toc530583161)

[8.1 Testes do modulo desktop 35](#_Toc530583162)

[8.2 Testes do modulo web 42](#_Toc530583163)

[8.3 Atualizações futuras 48](#_Toc530583164)

[9.0 Matérias do Semestre 49](#_Toc530583165)

[9.1 Programação Orientada a Objetos 2 49](#_Toc530583166)

[9.2 Tópicos especiais de programação orientada a objetos 49](#_Toc530583167)

[9.3 Desenvolvimento de Software para Internet 49](#_Toc530583168)

[9.4 Projeto Orientado a Objetos 49](#_Toc530583169)

[9.5 Gerenciamento de projeto de software 49](#_Toc530583170)

[10 Conclusão 50](#_Toc530583171)

[Referencias 51](#_Toc530583172)

[ANEXO 1 52](#_Toc530583173)

[ANEXO 2 118](#_Toc530583174)

[ANEXO 3 145](#_Toc530583175)

[ANEXO 4 147](#_Toc530583176)

# 1.0 Introdução

Neste semestre o tema do projeto integrado multidisciplinar, foi para que os alunos do terceiro e quarto semestre criassem um sistema de help-desk, o sistema será desenvolvido para uma empresa de prestação de serviços, que prestara serviços de suporte técnico para outras empresas através do sistema. O sistema Evolution, possui os recursos necessários para a administração de seus funcionários, clientes, dos chamados realizados e os produtos da empresa contratante. Através do sistema Evolution as empresas contratantes podem ter acesso a suporte para os produtos de sua empresa e podem contar com técnicos especializados para o serviço.

# 2.0 Projeto help desk Evolution

## 2.1 Objetivo

O objetivo do Sistema Evolution é possibilitar aos gestores terem um ambiente organizado e padronizado que permita a empresa identificar possíveis problemas no relacionamento de atendimento ao cliente. Analisado os problemas através do atendimento a distância focando no tempo para solução, as etapas realizadas até a solução, visando o grau de satisfação do cliente, e verificar meios de melhor atendimento e tomada de decisão.

Oferecendo um ambiente moderno para o profissional e para o cliente, podendo assim ampliar o relacionamento entre cliente e empresa.

Podendo também realizar uma avaliação dos colaboradores e possíveis problemas.

Além de arquivar informação dos atendimentos, estando disponíveis para a visualização.

## 2.2 Suposições e Restrições

Em seu desenvolvimento, consideramos alguns erros que os usuários poderiam cometer ao introduzir as informações, o sistema possui 3 níveis de usuário, sendo eles, coordenador, funcionário e cliente cada um com suas restrições. Para que um cliente possa abrir um atendimento ele deverá ter um login. O login poderá ter apenas um papel (Cliente ou Funcionário). Os usuários que possuírem o papel de Cliente terão acesso apenas a página web. Os usuários que possuírem o papel de Funcionário poderão ter o nível Coordenador ou Funcionário e terão acesso ao módulo desktop e página web. Quando o login tem o papel de funcionário, não poderá haver logins diferentes para o mesmo funcionário. Já para o cliente, poderá haver nomes de logins diferentes para o mesmo cliente.

## 2.3 Recursos

Para a produção do sistema Evolution foram usados softwares para montagem dos códigos, elaboração das interfaces e criação e administração do banco de dados, tais foram eles respectivamente, Visual Studio, Photoshop e MySQL Server 2012

Para a implementação do sistema foi primeiramente preciso a criação de um ambiente qualificado para a realização de tais tarefas, após a criação dos ambientes foi feito a aquisição dos equipamentos necessários e então pôr fim a implementação do servidor adequado para o funcionamento do sistema.

## 2.4 custos

Figura 1-Tabela de custos



Fonte:M.I.J.H Systens

Figura 2-Tabela de salario



Fonte:M.I.J.H Systens

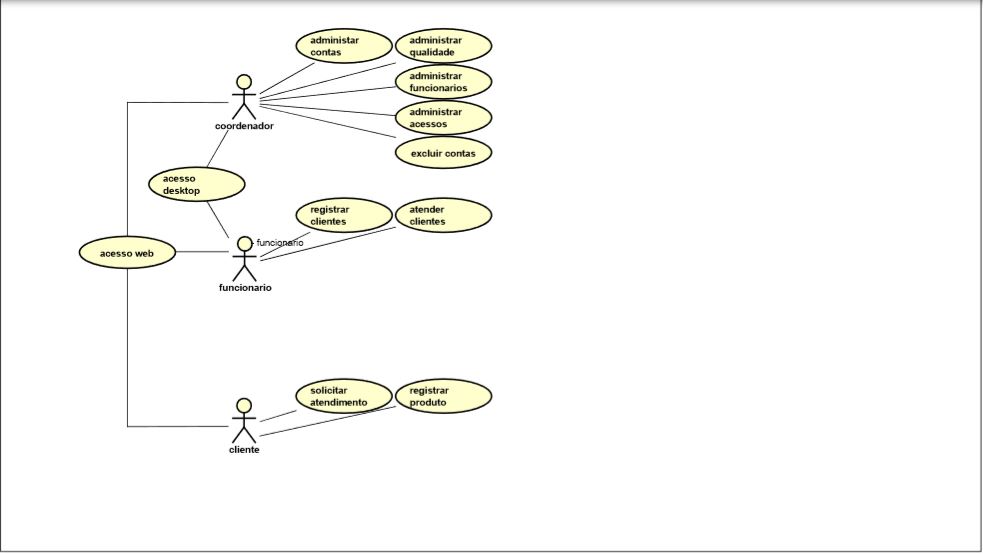
## 2.5 Datas das Fases e Interações das Linhas Base

Figura 3-Tabela de linha de base

|  |  |  |  |
| --- | --- | --- | --- |
| Datas das Fases e Iterações das Linhas Base | | | |
| Fase RUP | Iteração RUP | Linha Base | Data 05/06/2018  Alvo 15/07/2018 |
| Inception | Iniciação | Funcional | Data 25/07/2018 |
| Elaboration | Protótipo de  Arquitetura | Projeto | Data 15/08/2018  Alvo 01/09/2018 |
| Contruction | Release de Protótipo | Produto | Data 09/11/2018 |
| Transaction | Release Final | Produto | Data 13/11/2018 |

## 2.6 casos de uso

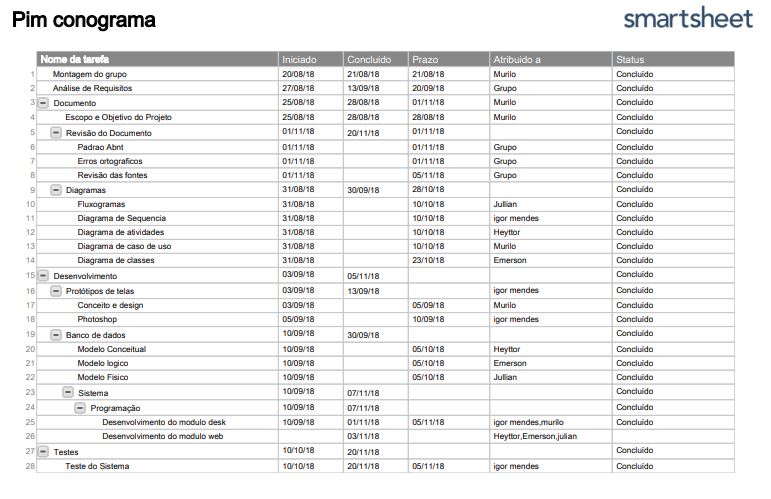
Figura 4-Casos de uso



Fonte:M.I.J.H Systens

## 2.7 Cronograma

Figura 5-Cronograma



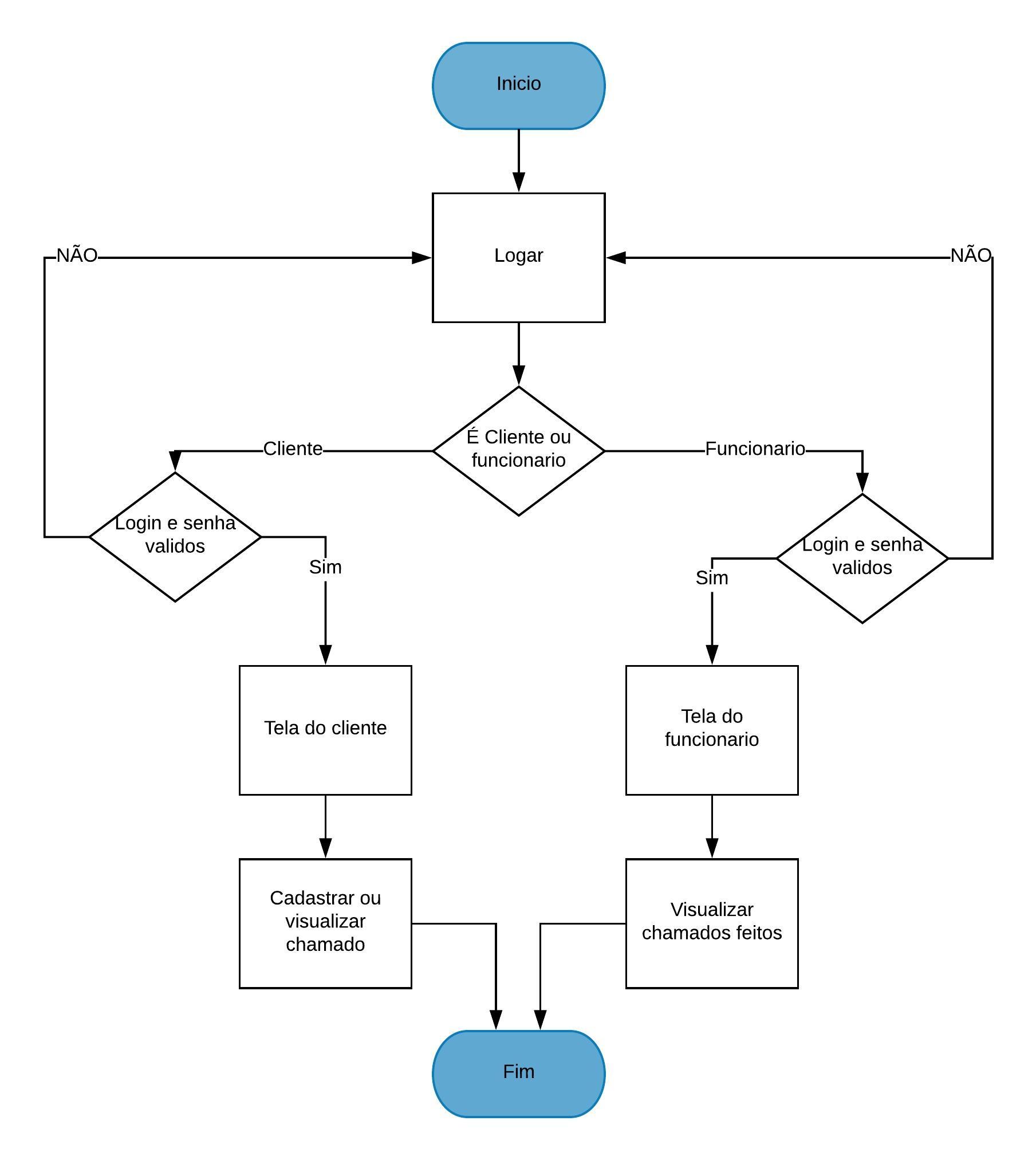
Fonte:M.I.J.H Systens

# 3.0 Sistema help desk Evolution

Este sistema foi desenvolvido em linguagens C# e ASP.NET, a linguagem C# foi utilizada no desenvolvimento da parte desktop do sistema que será a parte de administração onde somente os técnicos da empresa de suporte utilizarão, utilizando ASP.NET para a parte web do sistema que servira para uso do cliente, realizando chamadas para entrar em contato com o suporte técnico.

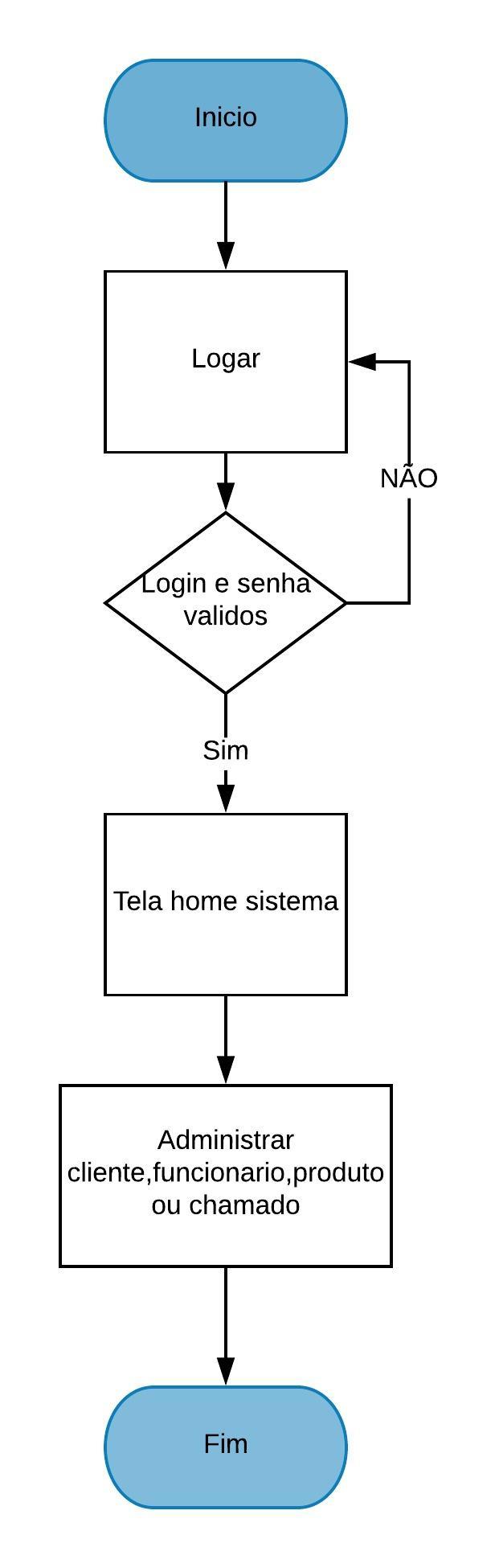
## 3.0.1 Fluxogramas

Figura 6-Fluxograma do modulo web



Fonte:M.I.J.H Systens

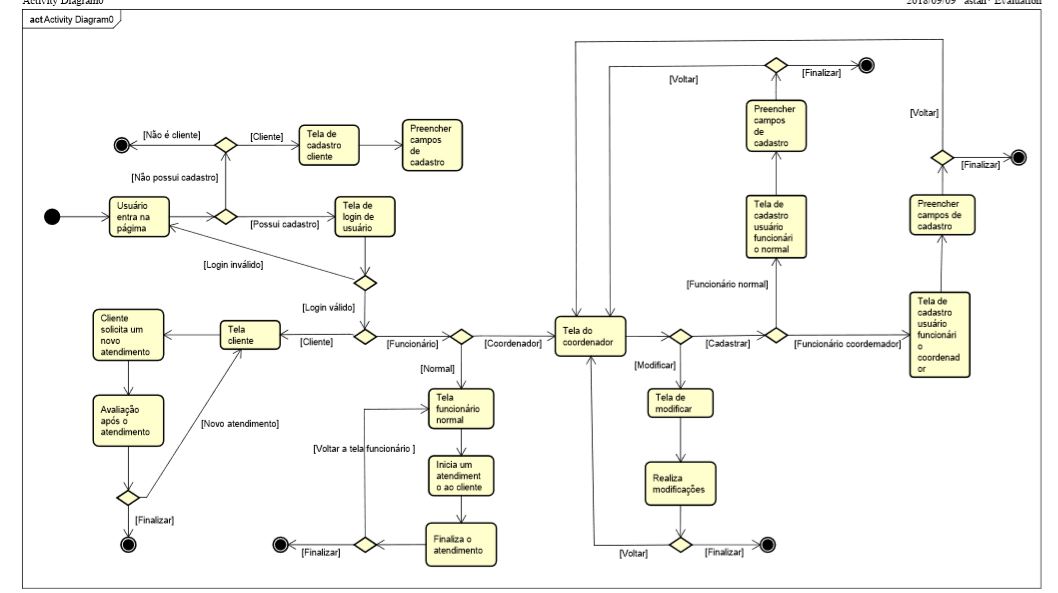
Figura 7-Fluxograma do modulo desktop



Fonte:M.I.J.H Systens

## 3.0.2 Diagrama de atividades

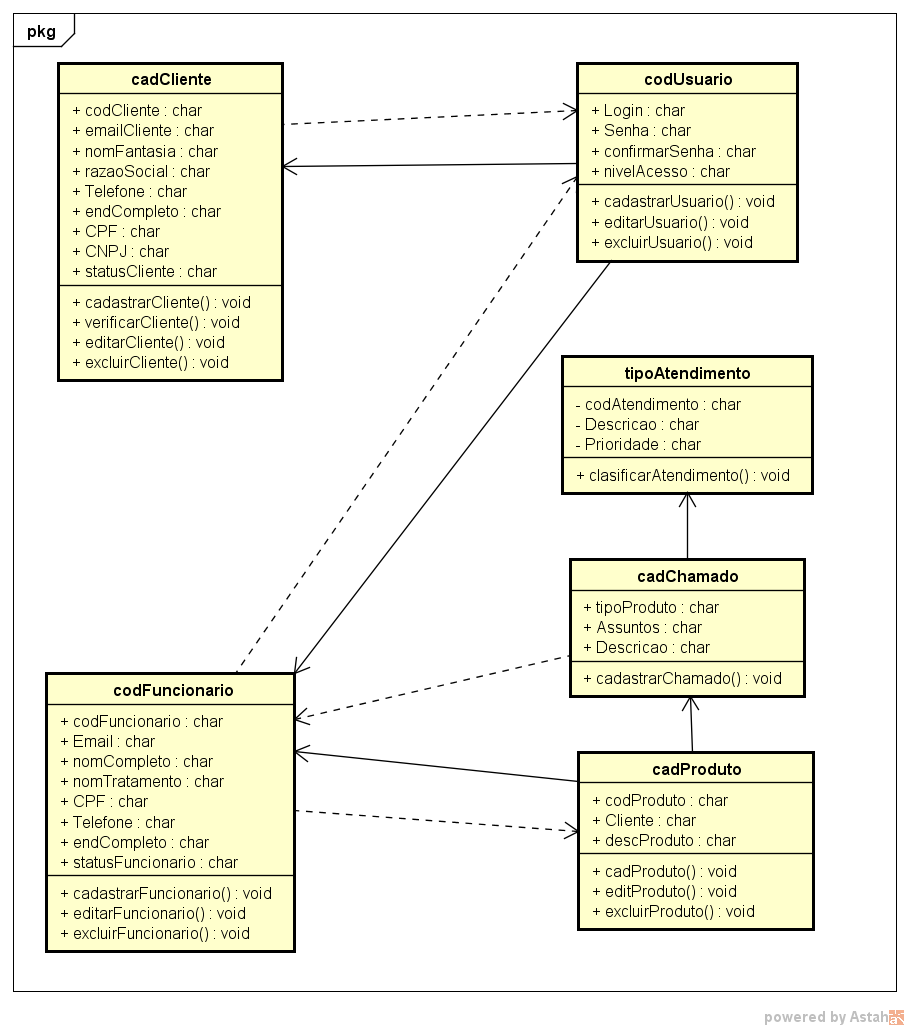
Figura 8-Diagrama de atividade do sistema



Fonte:M.I.J.H Systens

## 3.0.3 Diagrama de classes

Figura 9-Diagrama de classes



Fonte:M.I.J.H Systens

## 3.1 Desktop

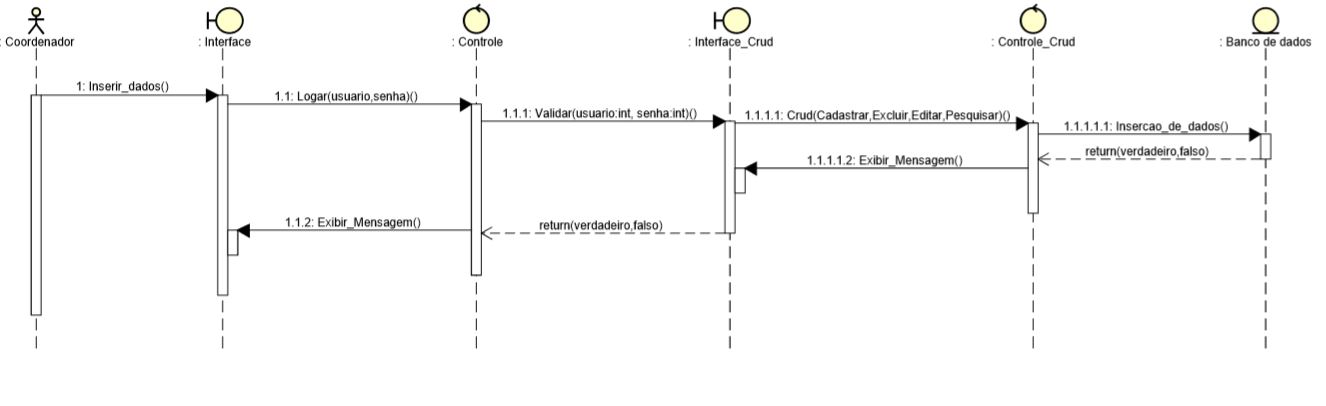
Este sistema foi desenvolvido em C# tanto na parte backend e frontend, utilizando o framework entity onde ele mapeia as classes e tabelas do banco, criando interação entre si

Foi utilizada a arquitetura MVC (model, view, controller), que divide a representação da informação do usuário com ele. É normalmente usado para o desenvolvimento de interface de usuário que divide uma aplicação em três parte interconectadas. Isto é feito para separar representações de informação internas dos modos como a informação é apresentada para e aceita pelo usuário.

A parte do sistema que ficara no desktop dos profissionais da empresa que prestara o serviço de suporte técnico. Onde somente o administrador e seus funcionários irão manuseá-los, este modulo terá a função de administrar o sistema, onde o administrador terá controle total sobre contas de clientes e funcionários, e o técnico terá controle sobre as contas dos clientes.

## 3.1.1 Diagrama de sequência

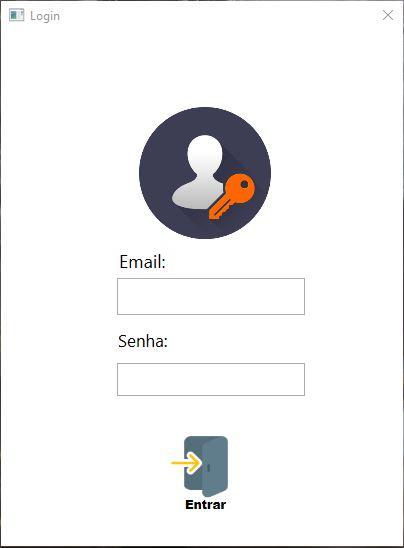
Figura 10-Diagrama de sequência do desktop



Fonte:M.I.J.H Systens

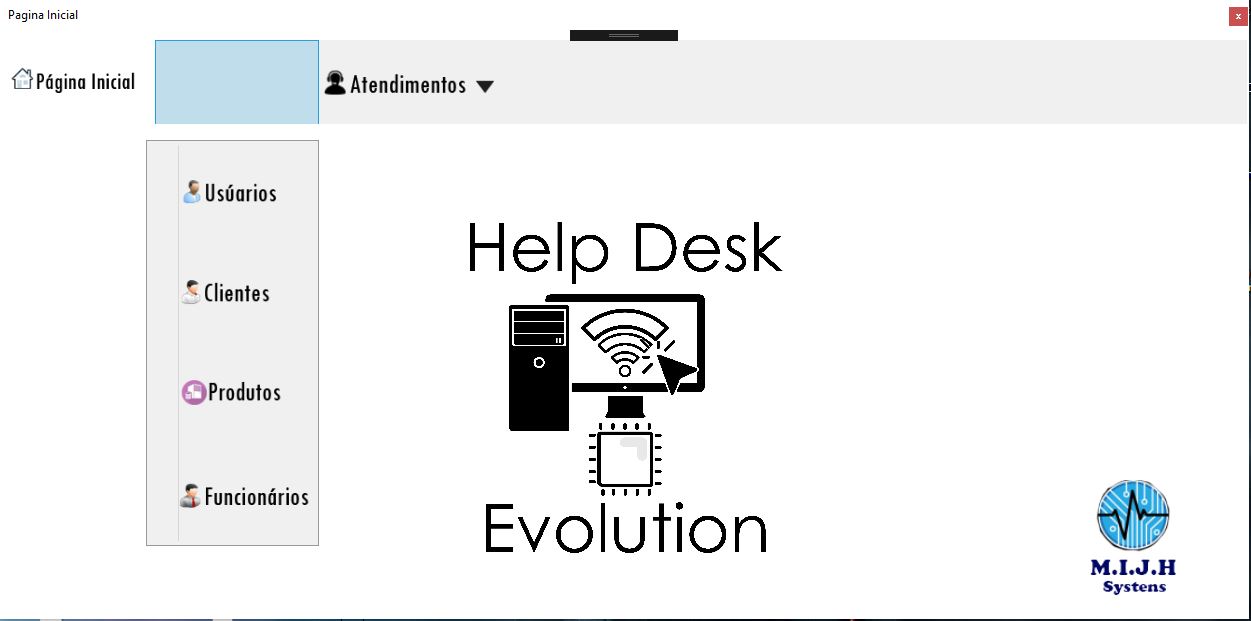
## 3.1.2 Layout de tela

Figura 11-Tela de login do modulo desktop



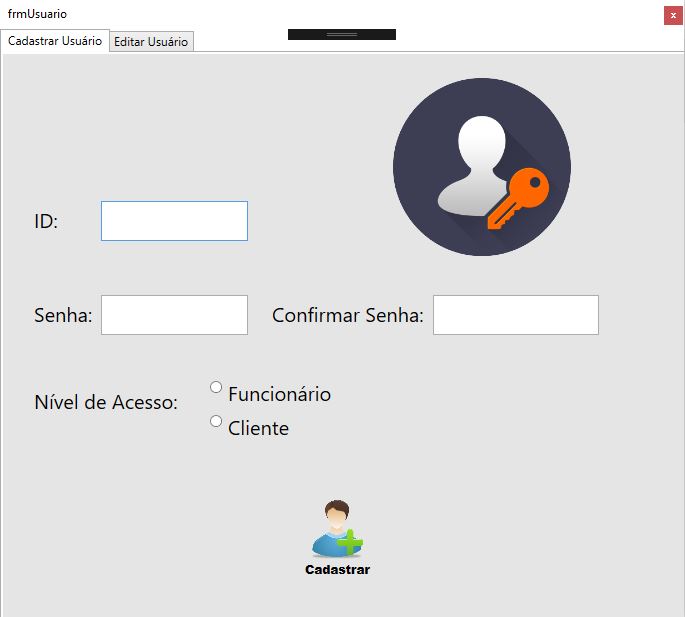
Fonte:M.I.J.H Systens

Figura 12-Tela home do desktop



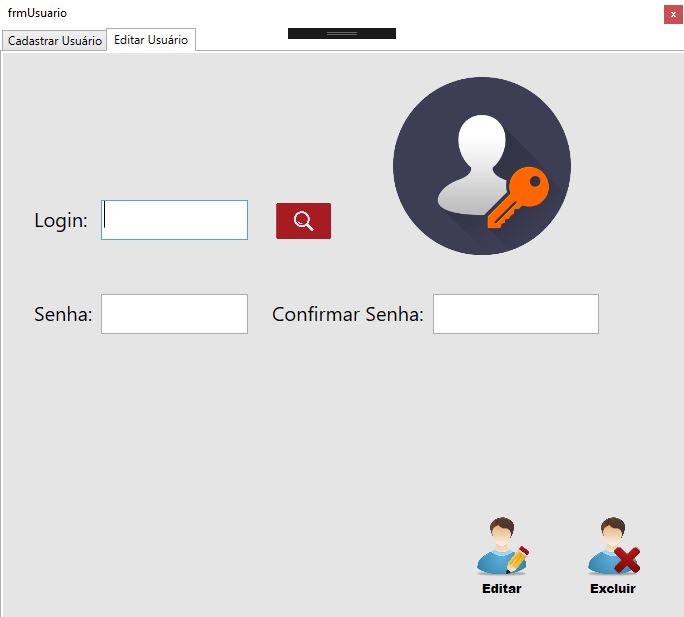
Fonte:M.I.J.H Systens

Figura 13-Tela de cadastrar usuário

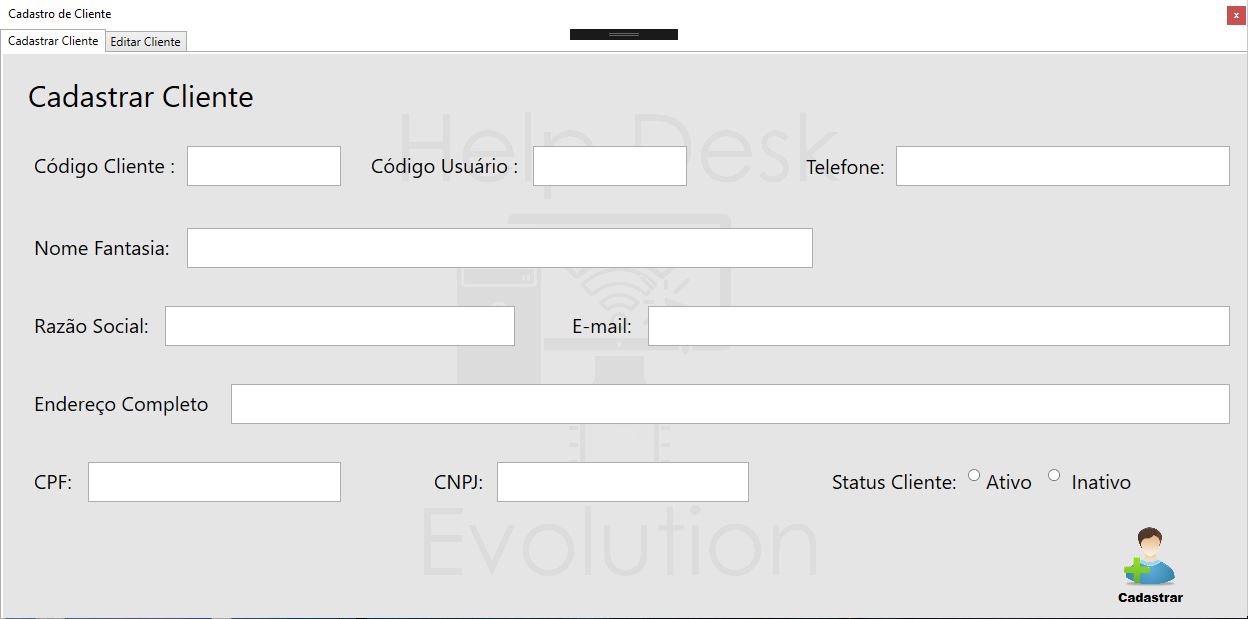


Fonte:M.I.J.H Systens

Figura 14-Tela de editar usuário

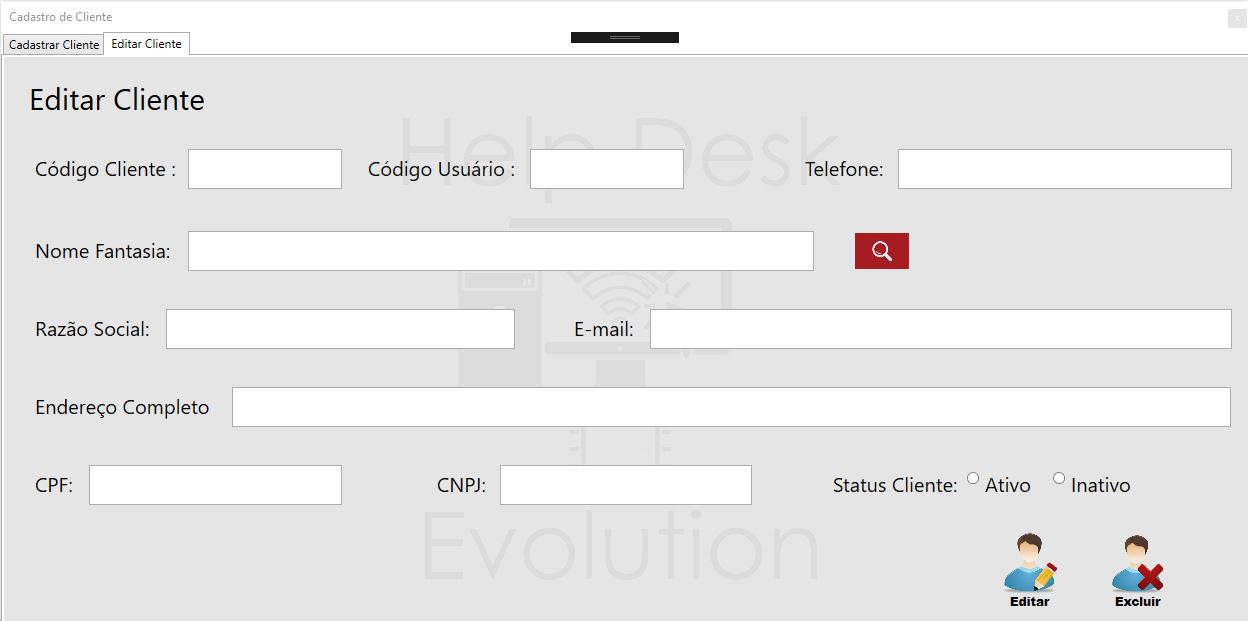


Fonte:M.I.J.H Systens



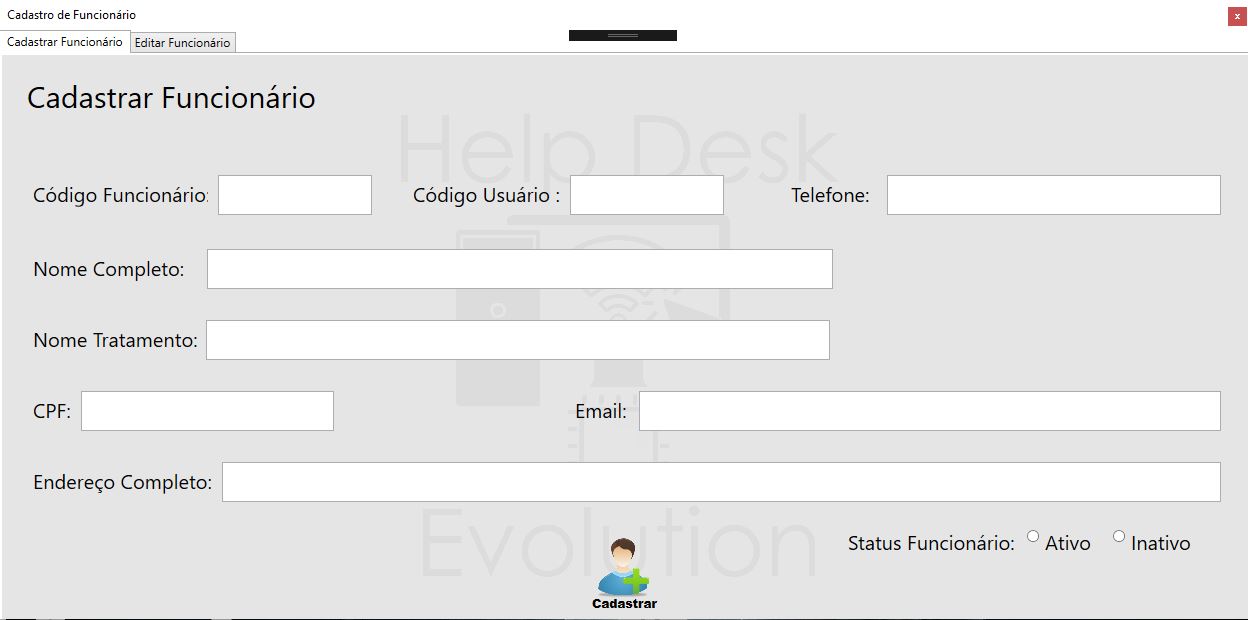
Fonte:M.I.J.H Systens

Figura 15-Tela de editar cliente



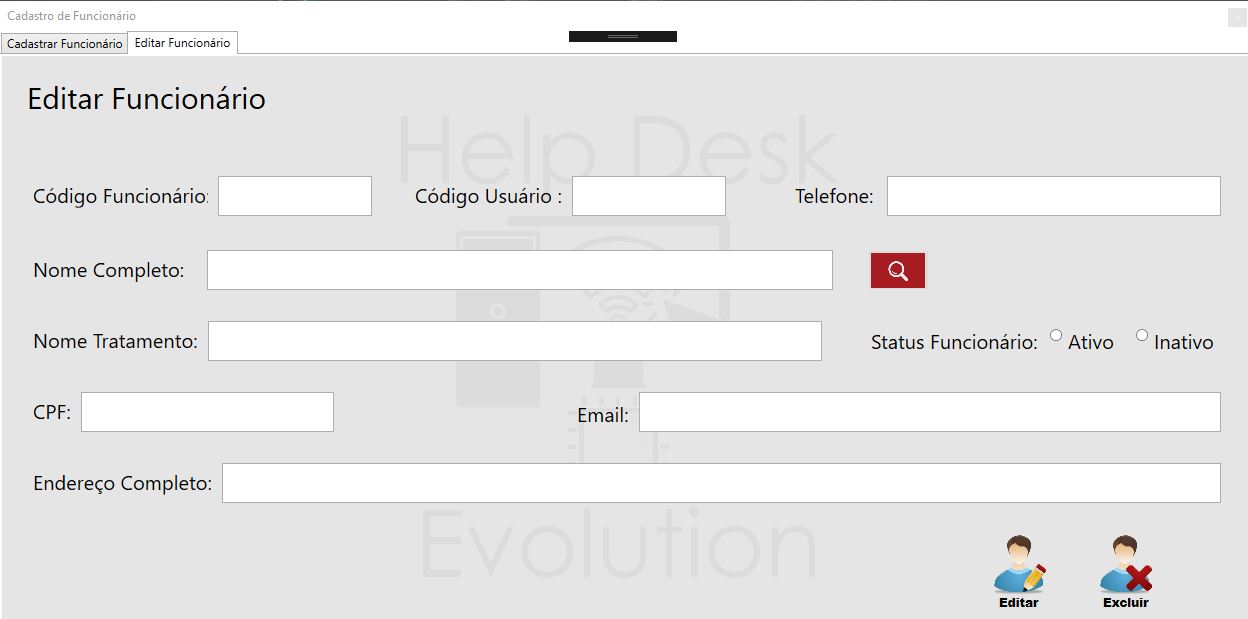
Fonte:M.I.J.H Systens

Figura 16-Tela de cadastrar funcionário



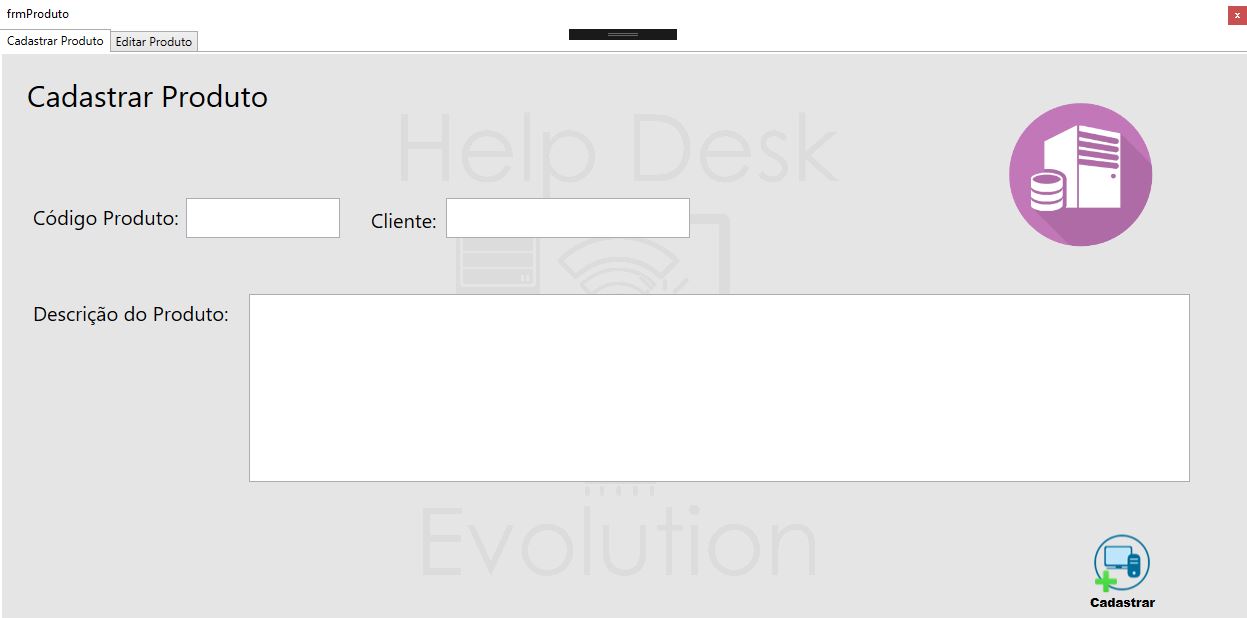
Fonte:M.I.J.H Systens

Figura 17-Tela de editar funcionário



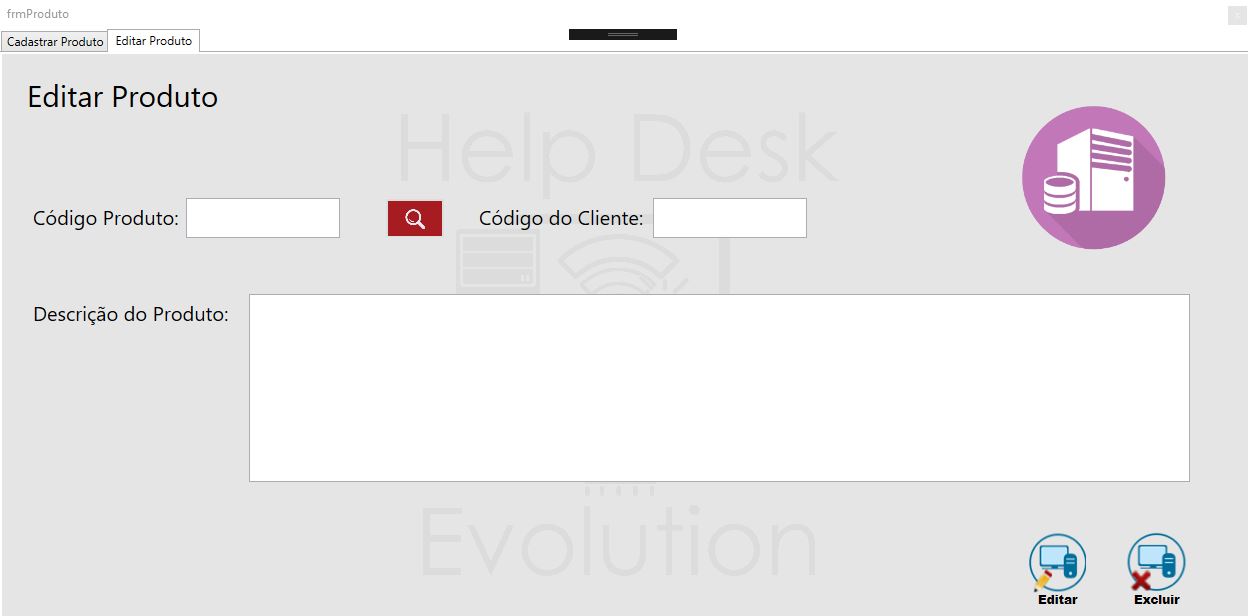
Fonte:M.I.J.H Systens

Figura 18-Tela de cadastrar produto



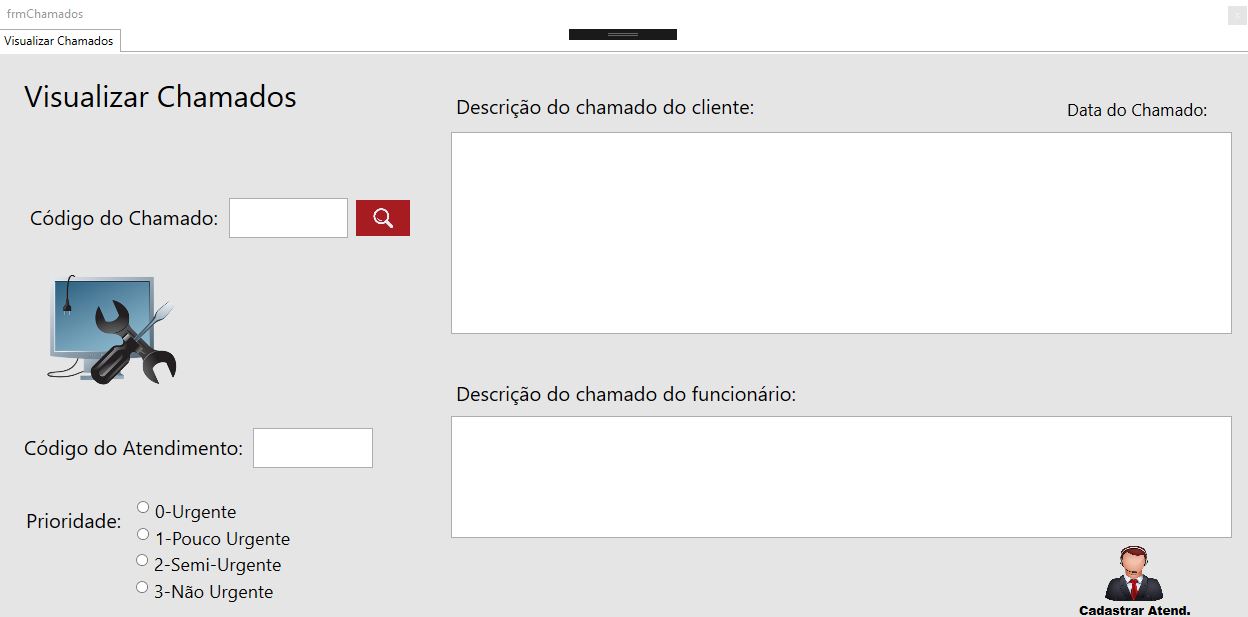
Fonte:M.I.J.H Systens

Figura 19-Tela de editar produto



Fonte:M.I.J.H Systens

Figura 20-Tela de cadastrar e visualizar atendimento



Fonte:M.I.J.H Systens

## 3.2 Web

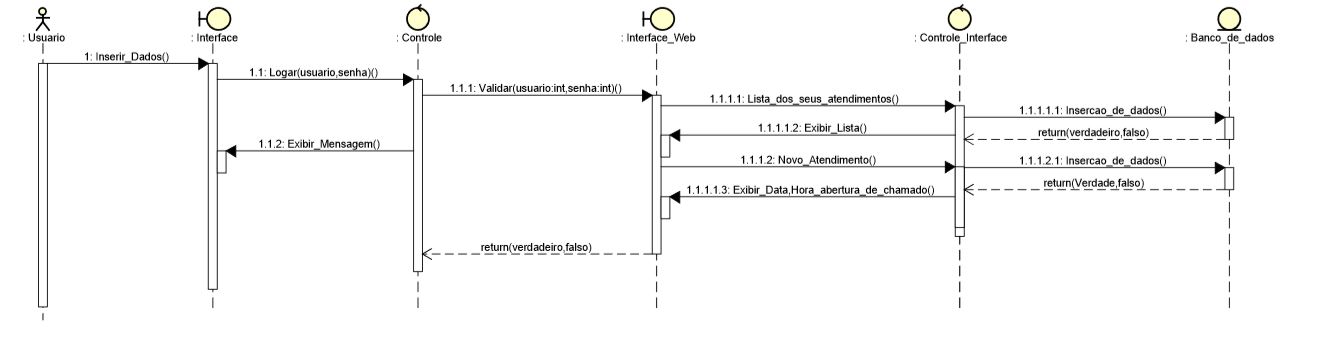
Este programa foi desenvolvido em ASP .Net, HTML, e JavaScript, toda a parte backend foi feita em ASP, utilizando o framework entity onde ele mape as classes e tabelas do banco, criando interação entre si em JavaScript foi utilizado jquery e Tooastr, para fazer a interação e comunicação das páginas, foi utilizado bootstrap para o HTML.

Foi utilizada a arquitetura MVC (model view controller), que divide a representação da informação do usuário com ele. É normalmente usado para o desenvolvimento de [interfaces de usuário](https://pt.wikipedia.org/wiki/Interface_do_usu%C3%A1rio) que divide uma aplicação em três partes interconectadas. Isto é feito para separar representações de informação internas dos modos como a informação é apresentada para e aceita pelo usuário.

Está é a parte web é aonde o cliente irá solicitar chamadas ao suporte, tudo por via web, este modulo terá a função para o cliente solicitar os serviços dos técnicos sobre um produto de sua empresa

## 3.2.1 Diagrama de sequência

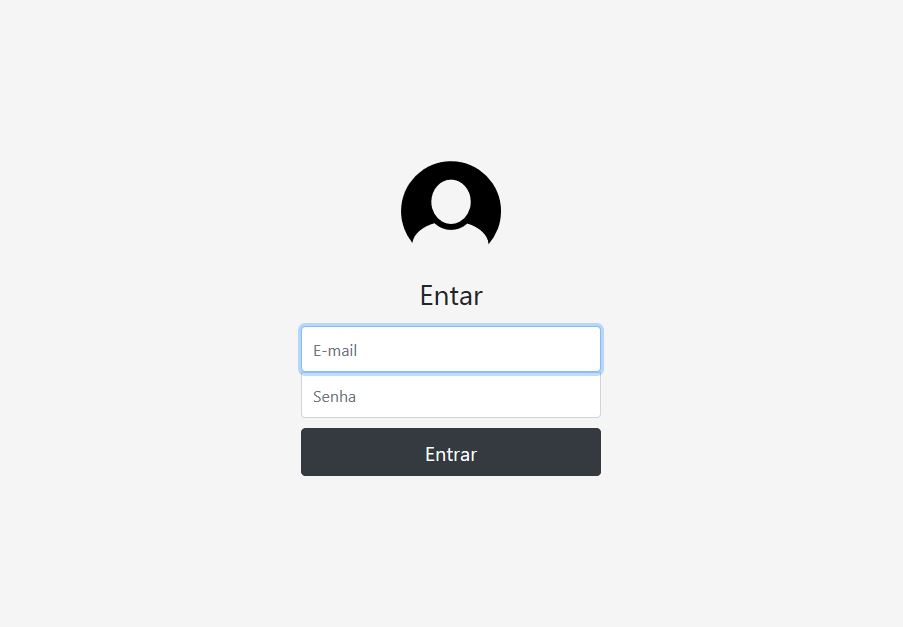
Figura 21-Diagrama de sequência do modulo web



Fonte:M.I.J.H Systens

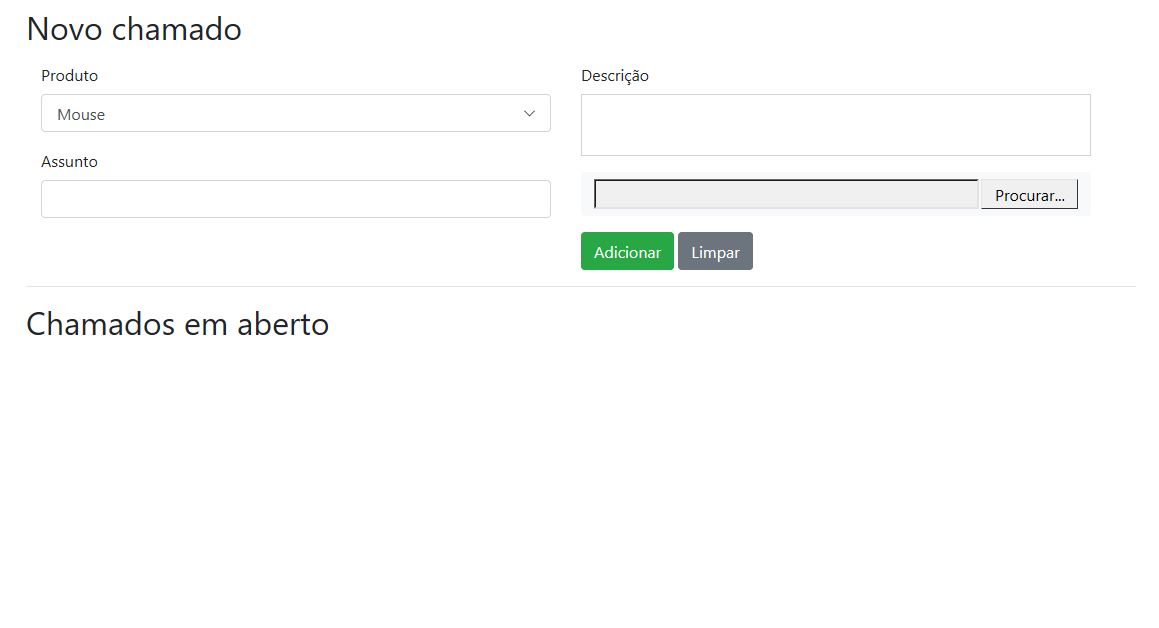
## 3.2.2 Layouts de telas

Figura 22-Tela do login do modulo web



Fonte:M.I.J.H Systens

Figura 23-Tela do cliente na web



Fonte:M.I.J.H Systens

Figura 24-Tela do funcionário na web



Fonte:M.I.J.H Systens

# 4.0 Desenvolvimento do projeto

Este projeto foi desenvolvido usando o programa Visual Studio, utilizando C# para a parte desktop e ASP.NET para a parte web.

## 4.1 Código da parte desktop

Os programas fontes desenvolvidos na parte desktop está presente no anexo 1 e no github (https://github.com/MurioLima/PIM-2018-2)

## 4.2 Código da parte web

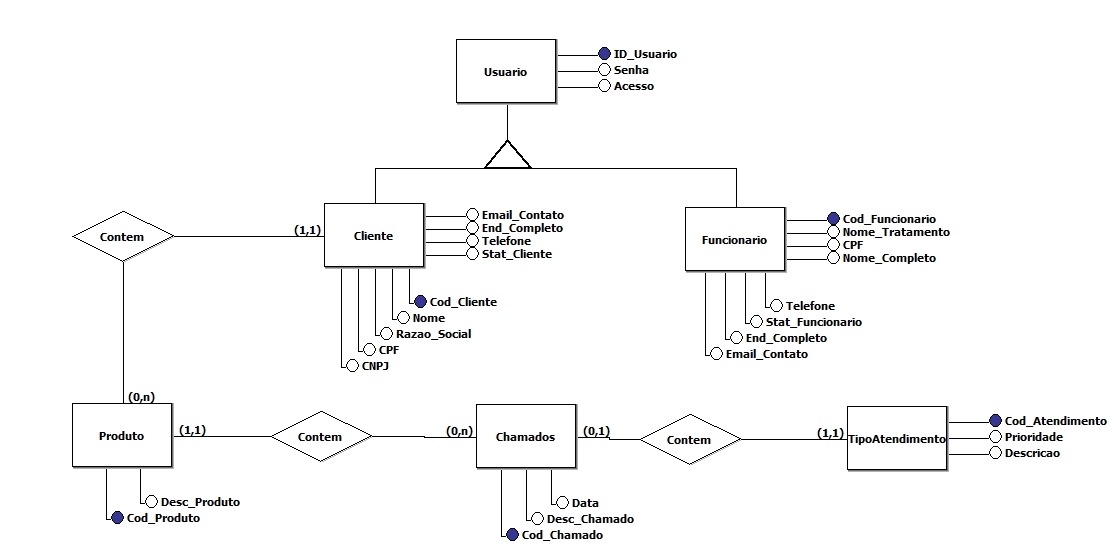
Os programas fontes desenvolvidos na parte web está presente no anexo 2 e no github (https://github.com/MurioLima/PIM-2018-2)

# 5.0 Banco de dados

Este sistema foi desenvolvido usando um banco de dados MySQL Server, e suas alterações e manutenção é feita através das ferramentas de SGBD do programa SQL Server Management Studio

## 5.1 Modelo Conceitual

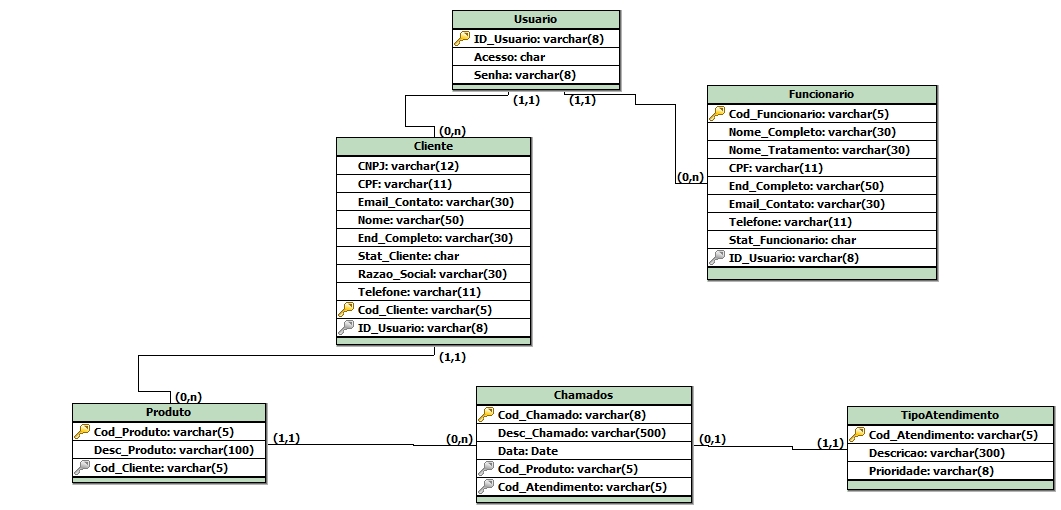
Figura 25-Modelo conceitual



Fonte:M.I.J.H Systens

## 5.2 Modelo Logico

Figura 26-Modelo logico



Fonte:M.I.J.H Systens

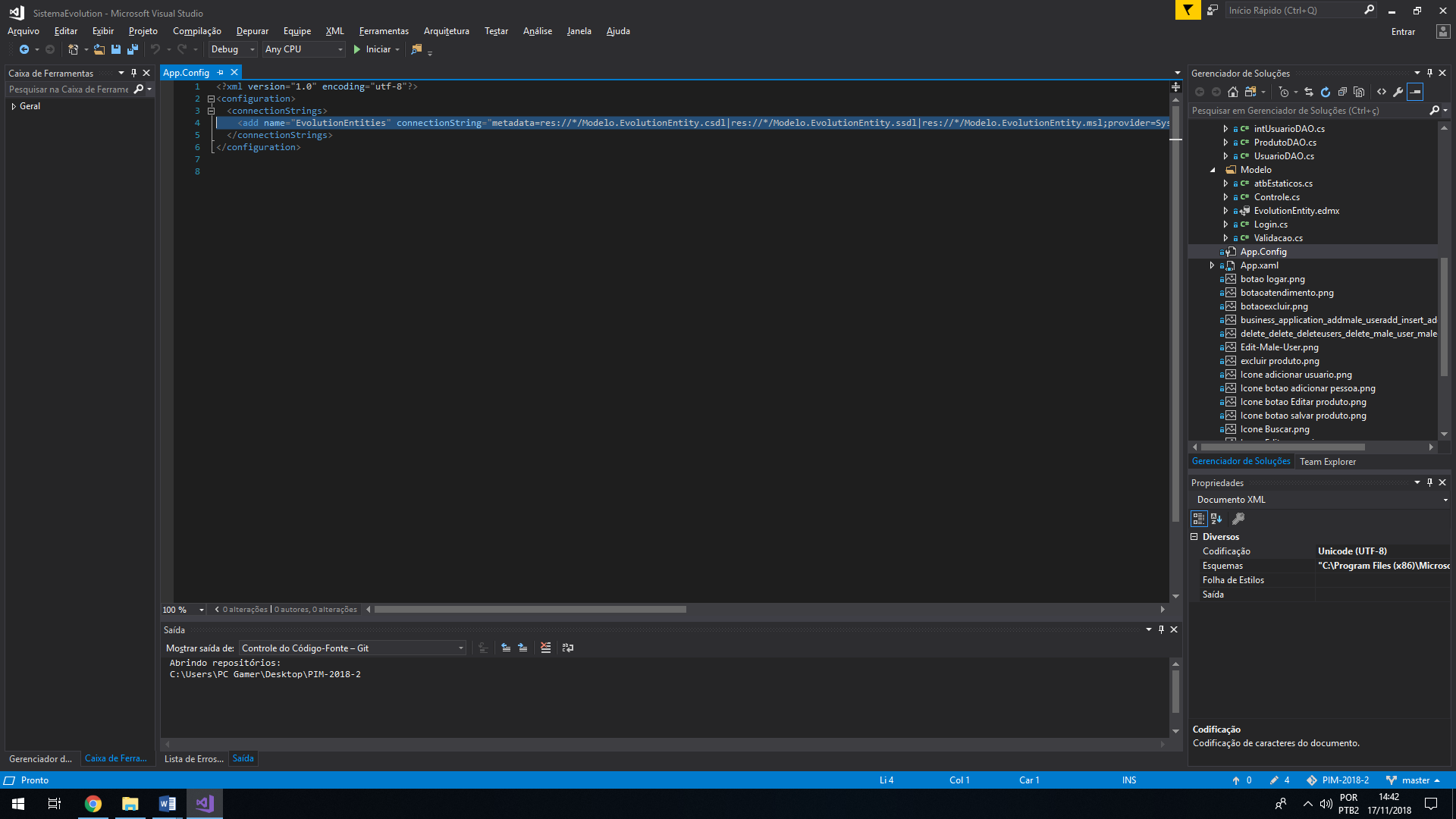
## 5.3 Modelo Físico

O Modelo Físico, ou seja, o Script do banco de dados usados no projeto Evolution está presente no anexo 3.

## 5.4 Conexões com o banco de dados

## 5.4.1 Conexão desktop

Figura 27-APP.Config



Fonte:M.I.J.H Systens

No arquivo App.Config há connectionString basta editar com as informações do seu banco:

Data Source = Endereço do banco

Catalog = Nome do banco de dados

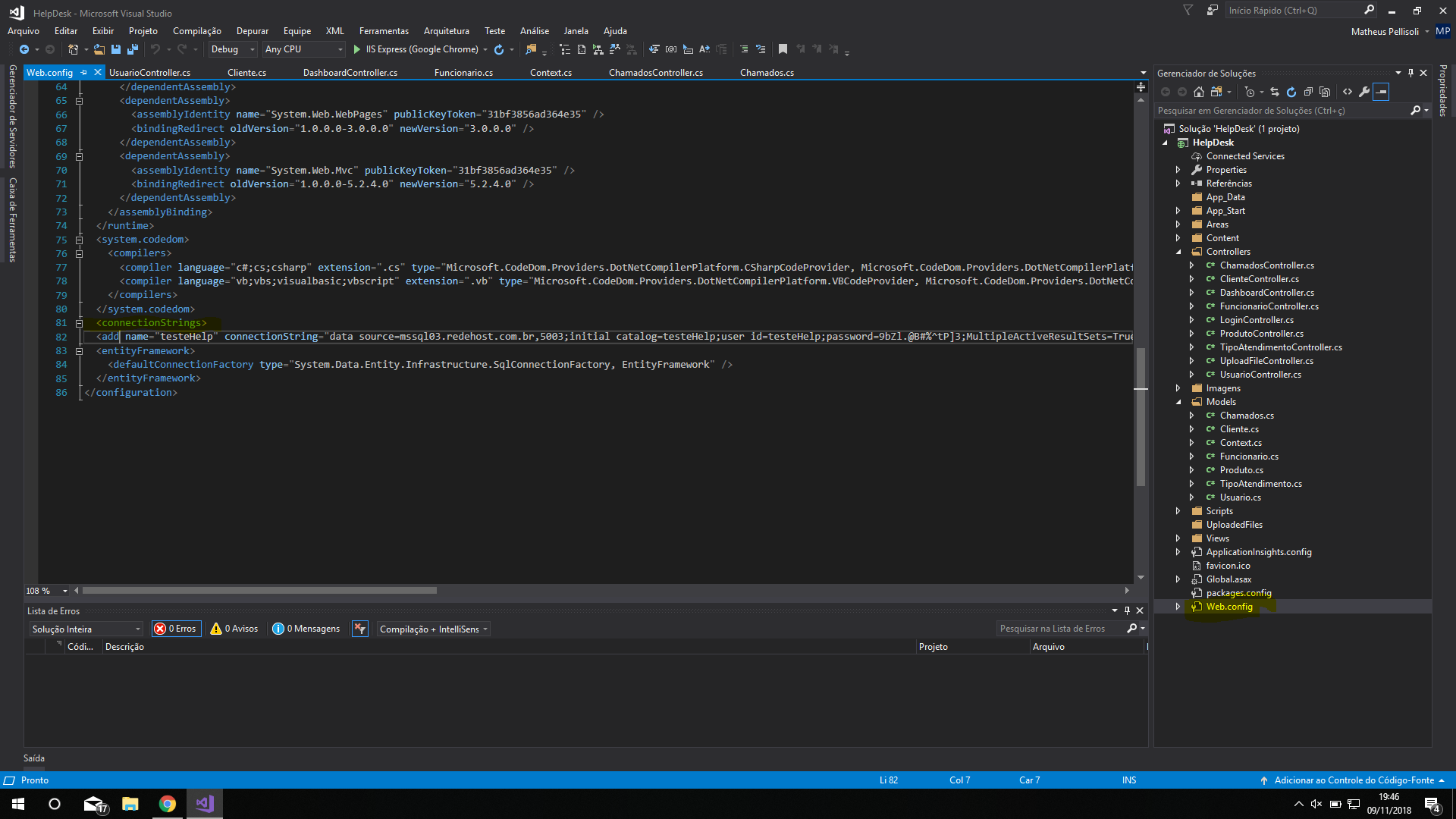
Id = Nome do usuário

Password = Senha do banco

Login utilizando autenticação do Windows = Integrated Security=True.

## 5.4.2 Conexão Web

Figura 28-WEB.Config



Fonte:M.I.J.H Systens

No arquivo Web.config há connectionString basta editar com as informações do seu banco

source = Endereço do banco

catalog = Banco de dados

id = nome do usuário

password = senha do banco

# 6.0 Requisitos de uso e desenvolvimento

Requisitos mínimos para o cliente e funcionário acessar o site web:

* Windows 7, Windows 8, Windows 8.1, Windows 10 ou versões posterior
* Processador Intel Pentium 4 ou posterior compatível com SSE2
* Processador de 32 bits (x86) ou 64 bits (x64) de 1 GHz ou superior\*
* 1 GB de RAM (32 bits) ou 2 GB de RAM (64 bits)
* 16 GB (32 bits) ou 20 GB (64 bits) de espaço em disco disponível

Requisitos mínimos para o desenvolvimento do sistema:

* Windows 10 versão 1507 ou superior: Home, Professional, Education e Enterprise (LTSC e S não são suportados)
* Windows Server 2016: Padrão e Datacenter
* Windows 8.1 (com [Atualização 2919355](https://support.microsoft.com/kb/2919355) ): Core, Professional e Enterprise
* Windows Server 2012 R2 (com a [atualização 2919355](https://support.microsoft.com/kb/2919355) ): Essentials, Standard, Datacenter
* Windows 7 SP1 (com as últimas atualizações do Windows): Home Premium, Professional, Enterprise, Ultimate.

Hardware:

* Processador de 1,8 GHz ou mais rápido. Dual-core ou melhor recomendado
* 2 GB de RAM; recomenda-se 4 GB de RAM (mínimo de 2,5 GB se for executado em uma máquina virtual)
* Espaço no disco rígido: até 130 GB de espaço disponível, dependendo dos recursos instalados; instalações típicas requerem 20 a 50 GB de espaço livre.
* Velocidade do disco rígido: para melhorar o desempenho, instale o Windows e o Visual Studio em uma unidade de estado sólido (SSD).
* Placa de vídeo que suporta uma resolução mínima de exibição de 720p (1280 por 720); O Visual Studio funcionará melhor em uma resolução de WXGA (1366 por 768) ou superior.

Banco de dados

* Requisitos para uso ou desenvolvimento
* Memória recomendado: Edições Express: 1 GB
* Qualquer outro tipo de memória deve conter no mínimo 4 GB e deve ter o aumento gradativo conforme o banco de dados cresce.
* Velocidade do processador recomendável: 2,0 GHz ou mais rápido
* Tipo de processador: processador x64, AMD Opteron, AMD Athlon 64, Intel Xeon com suporte Intel EM64T, Intel Pentium IV com suporte EM64T

# 7.0 Parametros de Qualidade

## 7.1 Funcionalidades

## 7.1.2 Adequação

O sistema Help-desk Evolution é possui dois módulos, um desktop para os funcionários, e um modulo web, feito para o uso dos clientes. O modulo desktop é onde acontece a administração de contas de clientes e funcionários, produtos e chamados, já o modulo web é feito somente para que o cliente solicite chamadas, portanto cada um dos módulos possui somente funcionalidades necessárias.

## 7.1.3 Acurácia

O sistema Help-desk Evolution é feito para prestar serviços de suporte técnico, os dados utilizados no sistema provêm de um banco de dados, ele exibe tais dados perfeitamente.

## 7.1.4 Interoperabilidade

O sistema Help-desk Evolution é um sistema único, com seu próprio banco de dados, que não possui interação com outros sistemas ou banco de dados.

## 7.1.5 Segurança

O sistema Help-desk Evolution, possui logins tanto para o módulo desktop quanto para o módulo web, o login do desktop possui uma checagem de dados de login e senha, para pessoas que não sejam funcionários possam logar , e o login da web também possui uma checagem de login e senha, para que clientes que não tenham um registro possam logar.

## 7.2 Confiabilidade

## 7.2.1 Maturidade

O sistema Help-desk Evolution é projetado para que não haja falhas decorrentes ao seu uso, e contém uma série de validações para que não ocorra erros relacionados os dados de seu banco de dados.

## 7.2.2 Tolerância a falhas

Não se aplica ao projeto.

## 7.2.3 Recuperabilidade

Caso haja algum tipo de falha com o sistema ele terá que ser reinstalado, para que não haja perda de dados, é recomendado que se tenha um backup do sistema e do banco de dados que esteja sendo utilizado.

## 7.3 Usabilidade

## 7.3.1 Inteligibilidade

O sistema Help-desk Evolution, em seus dois módulos foi desenvolvido com o objetivo de ser claro e intuitivo, tanto para os funcionários que trabalharam com desktops quanto para os clientes que acessarão a parte web, os seu layout é claro e fácil de compreender,

## 7.3.2 Apreensibilidade

O sistema Help-desk Evolution é de fácil aprendizado tanto para os funcionários que trabalharão com ele quanto para os clientes que o utilizaram.

## 7.4 Manutenibilidade

## 7.4.1 Analisabilidade

Os códigos do sistema Help-desk Evololution foram desenvolvidos com um padrão de orientação a objetos, para o modulo desktop foi desenvolvido com uma arquitetura de camadas, e o modulo web foi utilizado a arquitetura MVC, para que haja facilidade em análises futuras.

## 7.4.2 Modificabilidade

O sistema Help-desk Evolution foi desenvolvido em um padrão de orientação a objetos, para o modulo desktop foi desenvolvido com uma arquitetura de camadas, e o modulo web foi utilizado a arquitetura MVC, para facilitar mudanças futuras.

## 7.4.3 Estabilidade

O sistema Help-desk Evolution, foi desenvolvido para que opere com estabilidade, porem sua parte web só pode ser acessada pelo cliente caso haja uma conexão estável com a internet.

# 8.0 Testes

## 8.1 Testes do modulo desktop

**TESTE LOGIN SEM SUCESSO (CASO-1):**

Login sem sucesso por Erro da Senha.

**Objetivo do Teste:**

Testar o login de um usuário existente informando senha inválida para ele.

**Técnica utilizada:**

Digitação e análise dos resultados através da observação do testador.

**Estratégias:**

Utilizar o login do produto principal desenvolvido para ambiente desktop.

Utilizar a base de dados de testes.

**Ferramentas Necessárias:**

Computador PC e sistema operacional Windows 8.1 Pro ou 7 Pro.

Word para acesso a documentação.

**Condições de Teste:**

O usuário “igor@gmail.com” está cadastrado no banco com a senha “12345”

**Critério de Êxito**

Não permitir acessar o sistema com uma senha inválida para o usuário.

**Resultado:**

Satisfatório.

**Observações:** Embora o foco não tenha ficado como o previsto o objetivo do teste foi atingido, impedindo o login e limpando os campos para outra tentativa de login.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Procedimento de Teste | # | Resultado Esperado | Resultado Obtido |
| 1 | Na tela de login do sistema digitar:  Usuário: “Igor@gmail.com” Senha: “57458” | 2 | O sistema exibe uma mensagem: “Login ou Senha incorretos” | Mensagem exibida conforme o esperado |
| 3 | Clicar no botão [OK] da mensagem | 4 | O sistema exibe novamente a tela de login | Tela de login exibida conforme o esperado |

**Identificação do Testador:** Igor Mendes

Figura 29-Tabela de teste

Fonte:M.I.J.H Systens

**TESTE CADASTRAR CLIENTE SEM SUCESSO (CASO-2):**

Cadastrar cliente sem sucesso por inserir um código cliente existente.

**Objetivo do Teste:**

Testar o cadastramento de um cliente informando um código cliente já existente no banco de dados.

**Técnica utilizada:**

Digitação e análise dos resultados através da observação do testador.

**Estratégias:**

Utilizar a tela de cadastro do cliente desenvolvido para ambiente desktop.

Utilizar a base de dados de testes.

**Ferramentas Necessárias:**

Computador PC e sistema operacional Windows 8.1 Pro ou 7 Pro.

Word para acesso a documentação.

**Condições de Teste:**

O cliente “Mendes” está cadastrado no banco com o código cliente “2686”

**Critério de Êxito**

Não permitir cadastrar o novo cliente utilizando o mesmo código cliente.

**Resultado:**

Satisfatório.

**Observações:** O objetivo do teste foi atingido

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Procedimento de Teste | # | Resultado Esperado | Resultado Obtido |
| 1 | Na tela de cadastro do sistema digitar:  Código cliente: “2686”  Nome Fantasia: “Systems”  E-mail: “systems@gmail.com” | 2 | O sistema exibe uma mensagem: “Código do cliente já cadastrado, digite outro código.” | Mensagem exibida conforme o esperado |
| 3 | Clicar no botão [OK] da mensagem | 4 | O sistema exibe novamente a tela de cadastro para corrigir o código cliente. | Tela de cadastro exibida para corrigir o código cliente conforme o esperado. |

**Identificação do Testador**: Igor Mendes

Figura 30-Tabela de teste 2

Fonte:M.I.J.H Systens

**TESTE CADASTRAR FUNCIONÁRIO SEM SUCESSO (CASO-3):**

Cadastrar Funcionário sem sucesso por inserir um código funcionário existente.

**Objetivo do Teste:**

Testar o cadastramento de um funcionário informando um código funcionário já existente no banco de dados.

**Técnica utilizada:**

Digitação e análise dos resultados através da observação do testador.

**Estratégias:**

Utilizar a tela de cadastro do cliente desenvolvido para ambiente desktop.

Utilizar a base de dados de testes.

**Ferramentas Necessárias:**

Computador PC e sistema operacional Windows 8.1 Pro ou 7 Pro.

Word para acesso a documentação.

**Condições de Teste:**

O funcionário “Murilo” já está cadastrado no banco com o código cliente “888”.

**Critério de Êxito**

Não permitir cadastrar o novo funcionário utilizando o mesmo código funcionário.

**Resultado:**

Satisfatório.

**Observações:** O objetivo do teste foi atingido

**Identificação do Testador**: Igor Mendes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Procedimento de Teste | # | Resultado Esperado | Resultado Obtido |
| 1 | Na tela de cadastro do sistema digitar:  Código funcionário: “888”  Nome Completo: “Murilo”  E-mail: “murilo@gmail.com” | 2 | O sistema exibe uma mensagem: “Código do funcionário já cadastrado, digite outro código.” | Mensagem exibida conforme o esperado |
| 3 | Clicar no botão [OK] da mensagem | 4 | O sistema exibe novamente a tela de cadastro para corrigir o código funcionário. | Tela de cadastro exibida para corrigir o código funcionário conforme o esperado. |

Figura 31-Tabela de teste 3

Fonte:M.I.J.H Systens

**TESTE CADASTRAR PRODUTO SEM SUCESSO (CASO-4):**

Cadastrar Produto sem sucesso por inserir um código cliente não existente ou deixar vazio.

**Objetivo do Teste:**

Testar o cadastramento de um produto informando um código cliente não existente no banco de dados ou deixar vazio.

**Técnica utilizada:**

Digitação e análise dos resultados através da observação do testador.

**Estratégias:**

Utilizar a tela de cadastro do cliente desenvolvido para ambiente desktop.

Utilizar a base de dados de testes.

**Ferramentas Necessárias:**

Computador PC e sistema operacional Windows 8.1 Pro ou 7 Pro.

Word para acesso a documentação.

**Condições de Teste:**

O funcionário “Murilo” está cadastrado no banco com o código cliente “888”

**Critério de Êxito**

Não permitir cadastrar o novo produto por não informar um código cliente valido ou deixar vazio.

**Resultado:**

Satisfatório.

**Observações:** O objetivo do teste foi atingido

**Identificação do Testador**: Igor Mendes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Procedimento de Teste | # | Resultado Esperado | Resultado Obtido |
| 1 | Na tela de cadastro do sistema digitar:  Código Produto: “001”  Código Cliente: “0102”  Descrição Produto: “Mouse” | 2 | O sistema exibe uma mensagem: "Código do cliente já cadastrado ou vazio." | Mensagem exibida conforme o esperado |
| 3 | Clicar no botão [OK] da mensagem | 4 | O sistema exibe novamente a tela de cadastro para corrigir o código cliente. | Tela de cadastro exibida para corrigir o código cliente conforme o esperado. |

Figura 32-Tabela de teste 4

Fonte:M.I.J.H Systens

**CADASTRAR ATENDIMENTO SEM SUCESSO (CASO-5):**

Cadastrar atendimento sem sucesso por inserir um código de atendimento existente.

**Objetivo do Teste:**

Testar o cadastramento de um atendimento informando um código do atendimento existente no banco de dados.

**Técnica utilizada:**

Digitação e análise dos resultados através da observação do testador.

**Estratégias:**

Utilizar a tela de cadastro do cliente desenvolvido para ambiente desktop.

Utilizar a base de dados de testes.

**Ferramentas Necessárias:**

Computador PC e sistema operacional Windows 8.1 Pro ou 7 Pro.

Word para acesso a documentação.

**Condições de Teste:**

Um atendimento já está cadastrado no banco de dados com código do atendimento ”022”.

**Critério de Êxito**

Não permitir cadastrar um novo atendimento por informar um código do atendimento já existente no banco de dados.

**Resultado:**

Satisfatório.

**Observações:** O objetivo do teste foi atingido

**Identificação do Testador**: Igor Mendes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Procedimento de Teste | # | Resultado Esperado | Resultado Obtido |
| 1 | Na tela de chamados do sistema digitar:  Código do atendimento: “022”  Prioridade: “1”  Descrição: “Defeito no cabo” | 2 | O sistema exibe uma mensagem: "Código do Atendimento já cadastrado, digite outro código." | Mensagem exibida conforme o esperado |
| 3 | Clicar no botão [OK] da mensagem | 4 | O sistema exibe novamente a tela de chamados para corrigir o código atendimento. | Tela de chamados exibida para corrigir o código atendimento conforme o esperado. |

Figura 33-Tela de teste 5

Fonte:M.I.J.H Systens

|  |  |  |  |
| --- | --- | --- | --- |
| Identificação ou codificação: | | Nome do teste: Desktop teste | |
| Área a ser testada: Sistema desktop | | | |
| Técnica: Teste V&V | Data: 15/ 11/ 2018 | | Versão:1.0 |
| Objetivo: Testar o sistema desktop segundo as técnicas V&V. | | | |
| Ferramentas necessárias: um computador, Word para acesso a documentação. | | | |
| * **Teste de unidade:** Foi aplicado um total de 2 horas de testes procurando por erros de lógica e implementação em cada módulo, separadamente, validando dados válidos e inválidos via entrada/saída, os testes teve resultados positivos, toda a lógica do sistema se comportou como o projetado. * **Teste de integração:** Foi aplicado um total de 3 horas de testes com foco na integração entre as unidades do sistema, procurando falhas de envio e recebimento de dados, no tempo total do teste, não foi apresentado nenhuma falha neste requisito, todos os dados enviados ou recebidos do banco de dados se mostraram 100% eficaz. * **Teste de sistema:** O sistema foi testado no total de 6 horas de uso, todo o sistema funcionou 100% conforme o esperado, as telas se interagem corretamente, componentes são compatíveis, transferência de dados do sistema para o banco de dados se comportou-se muito bem, o resultado da avaliação foi totalmente positiva, dentro dos parâmetros esperados. * **Teste de segurança:** Foi feito uma serie de tentativas de login com senha ou usuário incorretos, o programa negou o acesso com êxito, pedindo o usuário verificar se está correta as informações. * **Teste de estresse**: Foi feito um teste de 5 funcionários acessando a parte desktop ao mesmo tempo, não ocorreu problemas, o teste foi básico pois raramente vários funcionários irão acessar o sistema ao mesmo tempo. * **Teste de desempenho:** No teste de desempenho foi feito o teste de velocidade em todas as telas e que se mostraram em uma média geral de para o envio e o recebimento dos dados que está dentro da margem de aceitação dos usuários, a velocidade em média é de 1s para o carregamento das telas e 2s para enviar e receber do banco de dados. * **Teste de caminho:** não se aplica. * **Teste de regressão:** não se aplica. * **Teste de caixa branca:** Foi aplicado umas séries de testes rigorosos em partes especificas examinando caminhos lógicos essenciais do sistema, verificando os componentes, exercitando os conjuntos específicos de condições e/ou ciclos do sistema, o código se mostrou totalmente executável nas considerações do analista. * **Teste de caixa preta:** não se aplica. | | | |
|  | | | |

Fonte:M.I.J.H Systens

## 8.2 Testes do modulo web

**TESTE LOGIN SEM SUCESSO (CASO-1):**

Login sem sucesso por Erro da Senha

**Objetivo do Teste:**

Testar o login de um usuário existente informando senha inválida para ele.

**Técnica utilizada:**

Digitação e análise dos resultados através da observação do testador.

**Estratégias:**

Utilizar o login do produto principal desenvolvido para ambiente web.

Utilizar a base de dados de testes.

**Ferramentas Necessárias:**

Computador PC, sistema operacional Windows 8.1 Pro ou 7 Pro e navegador de internet.

Word para acesso a documentação.

**Condições de Teste:**

O usuário “murilo@gmail.com” está cadastrado no banco com a senha “q12”

**Critério de Êxito**

Não permitir acessar o sistema com uma senha inválida para o usuário.

**Resultado:**

Satisfatório.

**Observações:** O objetivo do teste foi atingido

**Identificação do Testador**: Igor Mendes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Procedimento de Teste | # | Resultado Esperado | Resultado Obtido |
| 1 | Na tela de login do sistema digitar:  Usuário: “murilo@gmail.com” Senha: “q21q” | 2 | O sistema exibe uma mensagem: "Login ou Senha incorretos." e limpa os campos. | Os campos são limpos para outra tentativa de login porem a mensagem não é exibida. |

Figura 34-Tabela teste web

Fonte:M.I.J.H Systens

**TESTE LOGIN SEM INSERIR UM EMAIL VÁLIDO (CASO-2):**

Inserir um endereço inválido no login.

**Objetivo do Teste:**

Testar inserir um endereço de email sem ser um endereço de email.

**Técnica utilizada:**

Digitação e análise dos resultados através da observação do testador.

**Estratégias:**

Utilizar o login do produto principal desenvolvido para ambiente web.

Utilizar a base de dados de testes.

**Ferramentas Necessárias:**

Computador PC, sistema operacional Windows 8.1 Pro ou 7 Pro e navegador de internet e Word para acesso a documentação.

**Condições de Teste:**

Não necessita.

**Critério de Êxito**

Não permitir digitar um e-mail não valido.

**Resultado:**

Satisfatório.

**Observações:** O objetivo do teste foi atingido

**Identificação do Testador**: Igor Mendes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Procedimento de Teste | # | Resultado Esperado | Resultado Obtido |
| 1 | Na tela de login do sistema digitar:  Usuário: “Murilo.com” | 2 | O sistema exibe uma mensagem: "Você deve inserir um endereço de e-mail válido." e limpa os campos. | Mensagem exibida conforme o esperado |
| 3 | Mensagem desaparece ao apagar o campo incorreto. | 4 | O sistema permite novamente a inserção de um e-mail válido. | O sistema se comporta como o esperado. |

Figura 35-Tabela teste web 2

Fonte:M.I.J.H Systens

**TESTE LOGIN DEIXANDO O CAMPO SENHA EM BRANCO (CASO-3):**

Tentar fazer login sem inserir uma senha.

**Objetivo do Teste:**

Testar o login sem inserir um login.

**Técnica utilizada:**

Digitação e análise dos resultados através da observação do testador.

**Estratégias:**

Utilizar o login do produto principal desenvolvido para ambiente web.

Utilizar a base de dados de testes.

**Ferramentas Necessárias:**

Computador PC, sistema operacional Windows 8.1 Pro ou 7 Pro e navegador de internet.

Word para acesso a documentação.

**Condições de Teste:**

Não necessita.

**Critério de Êxito**

Não permitir acessar o sistema sem inserir uma senha.

**Resultado:**

Satisfatório.

**Observações:** O objetivo do teste foi atingido

**Identificação do Testador**: Igor Mendes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Procedimento de Teste | # | Resultado Esperado | Resultado Obtido |
| 1 | Na tela de login do sistema digitar:  Usuário: “murilo@gmail.com”  Senha:” “ | 2 | O sistema exibe uma mensagem: "Este campo é obrigatório”. | Mensagem exibida conforme o esperado |
| 3 | Mensagem desaparece após 2s | 4 | O sistema pede novamente a inserção da senha. | O sistema se comporta como o esperado. |

Figura 36-Tabela teste web 3

Fonte:M.I.J.H Systens

**CADASTRO DE CHAMADO DEIXANDO DESCRIÇÃO EM BRANCO (CASO-4):**

Tentar cadastrar um chamado deixando a descrição em branco.

**Objetivo do Teste:**

Testar o cadastramento do chamado deixando o campo descrição em branco.

**Técnica utilizada:**

Digitação e análise dos resultados através da observação do testador.

**Estratégias:**

Utilizar o login do produto principal desenvolvido para ambiente web.

Utilizar a base de dados de testes.

**Ferramentas Necessárias:**

Computador PC, sistema operacional Windows 8.1 Pro ou 7 Pro e navegador de internet.

Word para acesso a documentação.

**Condições de Teste:**

O produto “Mouse” está cadastrado no banco.

**Critério de Êxito**

Não permitir cadastrar o chamado.

**Resultado:**

Satisfatório.

**Observações:** O objetivo do teste foi atingido

**Identificação do Testador**: Igor Mendes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Procedimento de Teste | # | Resultado Esperado | Resultado Obtido |
| 1 | Na tela de chamados do sistema selecionar e digitar:  Produto: “Mouse”  Assunto: “ “  Descrição: “ “ | 2 | O sistema exibe uma mensagem: "Descrição necessária”. | Mensagem exibida conforme o esperado |
| 3 | Mensagem desaparece após 2s | 4 | O sistema pede novamente a inserção da descrição. | O sistema se comporta como o esperado. |

Figura 37-Tabela teste web 4

Fonte:M.I.J.H Systens

|  |  |  |  |
| --- | --- | --- | --- |
| Identificação ou codificação: | | Nome do teste: Web teste | |
| Área a ser testada: Sistema web | | | |
| Técnica: Teste V&V | Data: 17/ 11/ 2018 | | Versão:1.0 |
| Objetivo: Testar o sistema web segundo as técnicas V&V. | | | |
| Ferramentas necessárias: um computador, Word para acesso a documentação, navegador. | | | |
| * **Teste de unidade:** Foi aplicado um total de 1 hora de testes procurando por erros de lógica e implementação em cada módulo, separadamente, validando dados válidos e inválidos via entrada/saída, os testes teve resultados positivos, toda a lógica do sistema se comportou como o projetado. * **Teste de integração:** Foi aplicado um total de 2 horas de testes com foco na integração entre as unidades do sistema, procurando falhas de envio e recebimento de dados, no tempo total do teste, não foi apresentado nenhuma falha neste requisito, todos os dados enviados ou recebidos do banco de dados se mostraram 100% eficaz. * **Teste de sistema:** O sistema foi testado no total de 2 horas de uso, todo o sistema funcionou 100% conforme o esperado, as telas se interagem corretamente, componentes são compatíveis, transferência de dados do sistema para o banco de dados se comportou-se muito bem, o resultado da avaliação foi totalmente positiva, dentro dos parâmetros esperados. * **Teste de segurança:** Foi feito uma serie de tentativas de login com senha ou usuário incorretos, o programa negou o acesso com êxito, pedindo para o usuário verificar se está correta as informações. * **Teste de estresse**: Foi feito um teste de 5 clientes acessando a parte desktop ao mesmo tempo, não ocorreu problemas, o teste foi básico pois raramente um número maior de clientes irão acessar o sistema ao mesmo tempo. * **Teste de desempenho:** No teste de desempenho foi feito o teste de velocidade em todas as telas e que se mostraram em uma média geral de para o envio e o recebimento dos dados que está dentro da margem de aceitação dos usuários, a velocidade em média é de 1s para o carregamento das telas e 2s para enviar e receber do banco de dados. * **Teste de caminho:** não se aplica. * **Teste de regressão:** não se aplica. * **Teste de caixa branca:** Foi aplicado umas séries de testes rigorosos em partes especificas examinando caminhos lógicos essenciais do sistema, verificando os componentes, exercitando os conjuntos específicos de condições e/ou ciclos do sistema, o código se mostrou totalmente executável nas considerações do analista. * **Teste de caixa preta:** não se aplica. | | | |
|  | | | |

Fonte:M.I.J.H Systens

## 8.3 Atualizações futuras

|  |  |
| --- | --- |
| Sistema desktop | Sistema Web |
| Adicionar gráficos de quantidade de chamados resolvidos, aberto e encerrados para o funcionário. | Adicionar gráficos de quantidade de chamados resolvidos, aberto e encerrados para o cliente. |
| Adicionar um chat online no sistema web para o funcionário. | Adicionar um chat online no sistema web para o cliente. |
| Adicionar sistema de relatórios. | Adicionar visualização de relatórios. |
| Sistema de caixa de E-mail integrado com um webmail. | Sistema de caixa de E-mail integrado com um webmail. |
| Criptografia de senha. | Criptografia de senha. |

Figura 38-Tabela de Atualizações futuras

Fonte:M.I.J.H Systens

# 9.0 Matérias do Semestre

## 9.1 Programação Orientada a Objetos 2

A matéria de POO 2 continua a mostrar as técnicas da programação orientada a objetos, começando em como realmente trabalhar em um projeto usando apenas objetos, também ensinar em como trabalhar com banco de dados nos projetos , mostra como executar a conectividade com os bancos , ensina a linguagem sql,ensina como trabalhar com entity framework e as instruções em linq e gerenciamento de relatórios.

## 9.2 Tópicos especiais de programação orientada a objetos

A meteria de top esp de programação orientada a objetos foca no desenvolvimento de projetos em Java, e é introduzido o software android studio para a produção de aplicativos para dispositivos android, são ensinados as diferença entre as instruções de Java e C#, as ferramentas do android studio, funções de banco de dados do android studio e como desenvolver aplicativos com seus bancos de dados.

## 9.3 Desenvolvimento de Software para Internet

A matéria de desenvolvimento de software para internet, ensina o desenvolvimento de softwares para web, ensina as linguagens usadas para a programação web, mostra os conceitos e características dos softwares produzidos para web, ensina o conceitos da arquitetura MVC , e o desenvolvimento usando esta arquitetura, mostra os conceitos de API e como desenvolver usando APIS, mostra como utilizar pacotes como entity framework e web client e de como implementa-los no desenvolvimento dos projetos.

## 9.4 Projeto Orientado a Objetos

A meteria de Projeto Orientado a Objetos, ensina os conceitos do que é um projeto orientado a objetos, ensina os conhecimentos básicos sobre as metodologias e técnicas de projetos utilizados para o desenvolvimento de projetos, introduz as tecnologias de apoio para a produção de projetos.

## 9.5 Gerenciamento de projeto de software

A matéria de Gerenciamento de projeto de software, ensina os conceitos e técnicas de controle de projeto, ciclo de vida dos projetos, escopo, prazos e custos, gerenciamento de riscos e gestão de projetos de desenvolvimento de software.

# 10 Conclusão

De acordo com todos os processos de produção que foram utilizados para a conclusão do sistema Evolution, foi possível adquirir o conhecimento de como realmente são os processos de produção de um sistema na área de trabalho. Ao decorrer do desenvolvimento do sistema, foi possível ver como os componentes do curso se encaixam no meio de produção de um sistema.

No desenvolvimento do sistema Evolution, foi possível ver a utilização dos conteúdos de todas as matérias que foram dadas ao longo do curso em prática, começando com tudo que é relacionado ao gerenciamento de projetos e documentação, de como gerenciar um projeto com melhor eficiência, trabalhar com vários tipos de diagramas, fluxogramas, UML e seus artefatos, utilizando logica e técnicas de programação, aplicação de arquiteturas para melhor eficiência do sistema, gerenciamento de banco de dados, trabalhar com módulos em diferentes linguagens, para enfim desenvolver o sistema Help-Desk Evolution.

Com as pesquisas realizadas que complementaram nosso aprendizado no semestre, podemos observar que são várias atividades na execução de um projeto de sistema, tivemos bom conhecimento no assunto abordado no projeto e conseguimos produzir o produto final dentro do tempo estipulado pelo cliente, e com tudo isso podemos ver em pratica todo o conteúdo do curso de analise e desenvolvimento de sistema.

# Referencias

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. NBR ISO/IEC **9126-1 Engenharia de software - Qualidade de produto - Parte 1: Modelo de qualidade**. 2003.

Requisitos de sistema do Windows 7. *Disponível em<*https://support.microsoft.com/pt-br/help/10737/windows-7-system-requirements>*Acesso em: 20 nov. 2018.*

Requisitos de sistema do navegador Chrome. *Disponível em<*https://support.google.com/chrome/a/answer/7100626?hl=pt-BR>*Acesso em: 20 nov. 2018.*

 Visual Studio 2017 Product Family System Requirements. Disponivel em <<https://docs.microsoft.com/en-us/visualstudio/productinfo/vs2017-system-requirements-vs>> *Acesso em: 20 nov. 2018.*

# ANEXO 1

**Apresentacao**

**frmChamado**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace SistemaEvolution.Apresentacao

{

//Código para iniciar o form↓

public partial class frmChamado : Window

{

public frmChamado()

{

InitializeComponent();

}

//Código do botão para buscar o chamado↓

private void btnBuscarChamados\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaChamados = new List<string>();

ListaChamados.Add(txbCodigoChamado.Text);

ListaChamados.Add("");

Modelo.Controle controle = new Modelo.Controle();

controle.PesquisarChamados(ListaChamados);

Modelo.Chamados chamados = new Modelo.Chamados();

if (Modelo.atbEstaticos.listaChamadosEstatico == null)

{

MessageBox.Show("Nome Invalido");

}

else

if (Modelo.atbEstaticos.listaChamadosEstatico.Count() == 0)

{

MessageBox.Show("Não existe resposta para esta consulta");

}

else

if (Modelo.atbEstaticos.listaChamadosEstatico.Count() == 1)

{

chamados = Modelo.atbEstaticos.listaChamadosEstatico[0];

txbCodigoChamado.Text = chamados.Cod\_Chamado;

txbDescriçaoChamadoCliente.Text = chamados.Desc\_Chamado;

}

}

//Código do botão para cadastrar o chamado↓

private void Button\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaChamados = new List<string>();

List<String> ListaAtendimento = new List<string>();

ListaAtendimento.Add(txbCodigoAtendimento.Text);

ListaAtendimento.Add(txbDescricaoChamadoFuncionario.Text);

string Prioridade = "";

if (rdbUrgente.IsChecked == true) Prioridade = "0";

if (rdbPoucoUrgente.IsChecked == true) Prioridade = "1";

if (rdbSemiUrgente.IsChecked == true) Prioridade = "2";

if (rdbNaoUrgente.IsChecked == true) Prioridade = "3";

ListaAtendimento.Add(Prioridade);

Modelo.Controle controle = new Modelo.Controle();

controle.CadastrarTipoAtendimento(ListaAtendimento);

MessageBox.Show(controle.mensagem);

}

}

}

**frmClientes**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace SistemaEvolution.Apresentacao

{

/// <summary>

/// Interaction logic for frmClientes.xaml

/// </summary>

public partial class frmClientes : Window

{

//Exibição do form Cliente.

public frmClientes()

{

InitializeComponent();

}

private void RadioButton\_Checked(object sender, RoutedEventArgs e)

{

}

//Abaixo Codigo do botão Cadastrar cliente.

private void btnCadastrarUsuario\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaCliente = new List<string>();

ListaCliente.Add(txbCodCliente.Text);

ListaCliente.Add(txbNome.Text);

ListaCliente.Add(txbRazaoSocial.Text);

ListaCliente.Add(txbCpf.Text);

ListaCliente.Add(txbCnpj.Text);

ListaCliente.Add(txbEmail\_Contato.Text);

ListaCliente.Add(txbEndereço.Text);

ListaCliente.Add(txbTelefone.Text);

string Stat\_Cliente = "";

if (rdbStatusClienteAtivo.IsChecked == true) Stat\_Cliente = "A";

if (rdbStatusClienteInativo.IsChecked == true) Stat\_Cliente = "I";

ListaCliente.Add(Stat\_Cliente);

ListaCliente.Add(txbIdUsuario.Text);

Modelo.EvolutionEntities status = new Modelo.EvolutionEntities();

Modelo.Controle controle = new Modelo.Controle();

controle.CadastrarCliente(ListaCliente);

MessageBox.Show(controle.mensagem);

}

//Abaixo Codigo do botão Buscar pelo Id cliente.

private void btnBuscarCliente\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaCliente = new List<string>();

ListaCliente.Add(txbEDCodCliente.Text);

ListaCliente.Add("");

ListaCliente.Add("");

ListaCliente.Add("");

ListaCliente.Add("");

ListaCliente.Add("");

ListaCliente.Add("");

ListaCliente.Add("");

ListaCliente.Add("");

Modelo.Controle controle = new Modelo.Controle();

Modelo.Cliente cliente = controle.PesquisarCliente(ListaCliente);

if (controle.mensagem.Equals(""))

{

txbEDCodCliente.Text = cliente.Cod\_Cliente;

txbEDNome.Text = cliente.Nome;

txbEDRazaoSocial.Text = cliente.Razao\_Social;

txbEDCpf.Text = cliente.CPF;

txbEDCnpj.Text = cliente.CNPJ;

txbEDEmail\_Contato.Text = cliente.Email\_Contato;

txbEDEndereco.Text = cliente.End\_Completo;

txbEDTelefone.Text = cliente.Telefone;

txbEdIdUsuario.Text = cliente.ID\_usuario;

}

else

{

MessageBox.Show(controle.mensagem);

}

}

//Abaixo Codigo do botão Buscar pelo Nome cliente.

private void btnBuscarNome\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaCliente = new List<string>();

ListaCliente.Add("0");

ListaCliente.Add(txbEDNome.Text);

ListaCliente.Add("");

ListaCliente.Add("");

ListaCliente.Add("");

ListaCliente.Add("");

ListaCliente.Add("");

ListaCliente.Add("");

ListaCliente.Add("");

Modelo.Controle controle = new Modelo.Controle();

controle.PesquisarClientePorNome(ListaCliente);

Modelo.Cliente cliente = new Modelo.Cliente();

if(Modelo.atbEstaticos.listaClienteEstatico==null)

{

MessageBox.Show("Campo Nome está vazio");

}

else

if (Modelo.atbEstaticos.listaClienteEstatico.Count() ==0)

{

MessageBox.Show("Não existe resposta para esta consulta");

}

else

if (Modelo.atbEstaticos.listaClienteEstatico.Count() == 1)

{

cliente = Modelo.atbEstaticos.listaClienteEstatico[0];

txbEDCodCliente.Text = cliente.Cod\_Cliente;

txbEDNome.Text = cliente.Nome;

txbEDRazaoSocial.Text = cliente.Razao\_Social;

txbEDCpf.Text = cliente.CPF;

txbEDCnpj.Text = cliente.CNPJ;

txbEDEmail\_Contato.Text = cliente.Email\_Contato;

txbEDEndereco.Text = cliente.End\_Completo;

txbEDTelefone.Text = cliente.Telefone;

txbEdIdUsuario.Text = cliente.ID\_usuario;

}

else

if (Modelo.atbEstaticos.listaClienteEstatico.Count >= 2)

{

frmSelecao frmS = new frmSelecao();

frmS.ShowDialog();

txbEDCodCliente.Text = Modelo.atbEstaticos.listaClienteEstatico[0].Cod\_Cliente.ToString();

txbEDNome.Text = Modelo.atbEstaticos.listaClienteEstatico[0].Nome.ToString();

txbEDRazaoSocial.Text = Modelo.atbEstaticos.listaClienteEstatico[0].Razao\_Social.ToString();

txbEDCpf.Text = Modelo.atbEstaticos.listaClienteEstatico[0].CPF.ToString();

txbEDCnpj.Text = Modelo.atbEstaticos.listaClienteEstatico[0].CNPJ.ToString();

txbEDEmail\_Contato.Text = Modelo.atbEstaticos.listaClienteEstatico[0].Email\_Contato.ToString();

txbEDEndereco.Text = Modelo.atbEstaticos.listaClienteEstatico[0].End\_Completo.ToString();

txbEDTelefone.Text = Modelo.atbEstaticos.listaClienteEstatico[0].Telefone.ToString();

txbEdIdUsuario.Text = Modelo.atbEstaticos.listaClienteEstatico[0].ID\_usuario.ToString();

}

}

//Abaixo Codigo do botão Excluir cliente.

private void btnExcluirCliente\_Click(object sender, RoutedEventArgs e)

{

String[] dados = { txbEDCodCliente.Text, txbEDNome.Text, txbEDRazaoSocial.Text, txbEDCpf.Text, txbEDCnpj.Text, txbEDEmail\_Contato.Text, txbEDEndereco.Text, txbEDTelefone.Text };

List<String> ListaCliente = new List<string>(dados);

Modelo.Controle controle = new Modelo.Controle();

MessageBoxResult resultado = MessageBox.Show("Deseja realmente excluir ?",

"Alerta de exclusão", MessageBoxButton.YesNo, MessageBoxImage.Warning);

if (resultado == MessageBoxResult.Yes)

{

controle.ExcluirCliente(ListaCliente);

MessageBox.Show(controle.mensagem);

}

}

//Abaixo Codigo do botão Editar cliente.

private void btnEditarUsuario\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaCliente = new List<string>();

ListaCliente.Add(txbEDCodCliente.Text);

ListaCliente.Add(txbEDNome.Text);

ListaCliente.Add(txbEDRazaoSocial.Text);

ListaCliente.Add(txbEDCpf.Text);

ListaCliente.Add(txbEDCnpj.Text);

ListaCliente.Add(txbEDEmail\_Contato.Text);

ListaCliente.Add(txbEDEndereco.Text);

ListaCliente.Add(txbEDTelefone.Text);

string Stat\_Cliente = "";

if (rdbEDStatusClienteAtivo.IsChecked == true) Stat\_Cliente = "A";

if (rdbEDStatusFuncionarioInativo.IsChecked == true) Stat\_Cliente = "I";

ListaCliente.Add(Stat\_Cliente);

ListaCliente.Add(txbEdIdUsuario.Text);

Modelo.Controle controle = new Modelo.Controle();

controle.EditarCliente(ListaCliente);

MessageBox.Show(controle.mensagem);

Modelo.EvolutionEntities status = new Modelo.EvolutionEntities();

}

private void rdbStatusClienteInativo\_Checked(object sender, RoutedEventArgs e)

{

}

}

}

**frmFuncionario**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace SistemaEvolution.Apresentacao

{

/// <summary>

/// Interaction logic for frmFuncionario.xaml

/// </summary>

public partial class frmFuncionario : Window

{

//Código para iniciar o form↓

public frmFuncionario()

{

InitializeComponent();

}

//Código do botao para chamar o cadastro funcionario↓

private void btnCadastrarFuncionario\_Click\_1(object sender, RoutedEventArgs e)

{

List<String> ListaFuncionario = new List<string>();

ListaFuncionario.Add(txbCodFuncionario.Text);

ListaFuncionario.Add(txbNomeCompleto.Text);

ListaFuncionario.Add(txbNomeTratamento.Text);

ListaFuncionario.Add(txbCpf.Text);

ListaFuncionario.Add(txbEndereço.Text);

ListaFuncionario.Add(txbTelefone.Text);

ListaFuncionario.Add(txbEmailContato.Text);

string stat\_Funcionario = "";

if (rdbStatusFuncionarioAtivo.IsChecked == true) stat\_Funcionario = "A";

if (rdbStatusFuncionarioInativo.IsChecked == true) stat\_Funcionario = "I";

ListaFuncionario.Add(stat\_Funcionario);

ListaFuncionario.Add(txbIdUsuario.Text);

Modelo.Controle controle = new Modelo.Controle();

controle.CadastrarFuncionario(ListaFuncionario);

MessageBox.Show(controle.mensagem);

}

//Código do botao para chamar o buscar pelo nome funcionario↓

private void btnBuscarNome\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaFuncionario = new List<string>();

ListaFuncionario.Add("0");

ListaFuncionario.Add(txbEDNomeCompleto.Text);

ListaFuncionario.Add("");

ListaFuncionario.Add("");

ListaFuncionario.Add("");

ListaFuncionario.Add("");

ListaFuncionario.Add("");

ListaFuncionario.Add("");

Modelo.Controle controle = new Modelo.Controle();

controle.PesquisarFuncionarioPorNome(ListaFuncionario);

Modelo.Funcionario funcionario = new Modelo.Funcionario();

if (Modelo.atbEstaticos.listaFuncionarioEstatico == null)

{

MessageBox.Show("Campo Nome está vazio");

}

else

if (Modelo.atbEstaticos.listaFuncionarioEstatico.Count() == 0)

{

MessageBox.Show("Não existe resposta para esta consulta");

}

else

if (Modelo.atbEstaticos.listaFuncionarioEstatico.Count() == 1)

{

funcionario = Modelo.atbEstaticos.listaFuncionarioEstatico[0];

txbEDCodFuncionario.Text = funcionario.Cod\_Funcionario;

txbEDNomeCompleto.Text = funcionario.Nome\_Completo;

txbEDNomeTratamento.Text = funcionario.Nome\_Tratamento;

txbEDCpf.Text = funcionario.CPF;

txbEDEndereco.Text = funcionario.End\_Completo;

txbEDTelefone.Text = funcionario.Telefone;

txbEDEmailContato.Text = funcionario.Email\_Contato;

txbEdIdUsuario.Text = funcionario.ID\_usuario;

}

else

if (Modelo.atbEstaticos.listaFuncionarioEstatico.Count >= 2)

{

frmSelecaoFuncionario frmS = new frmSelecaoFuncionario();

frmS.ShowDialog();

txbEDCodFuncionario.Text = Modelo.atbEstaticos.listaFuncionarioEstatico[0].Cod\_Funcionario.ToString();

txbEDNomeCompleto.Text = Modelo.atbEstaticos.listaFuncionarioEstatico[0].Nome\_Completo.ToString();

txbEDNomeTratamento.Text = Modelo.atbEstaticos.listaFuncionarioEstatico[0].Nome\_Tratamento.ToString();

txbEDCpf.Text = Modelo.atbEstaticos.listaFuncionarioEstatico[0].CPF.ToString();

txbEDEndereco.Text = Modelo.atbEstaticos.listaFuncionarioEstatico[0].End\_Completo.ToString();

txbEDTelefone.Text = Modelo.atbEstaticos.listaFuncionarioEstatico[0].Telefone.ToString();

txbEDEmailContato.Text = Modelo.atbEstaticos.listaFuncionarioEstatico[0].Email\_Contato.ToString();

txbEdIdUsuario.Text = Modelo.atbEstaticos.listaFuncionarioEstatico[0].ID\_usuario.ToString();

}

}

//Código do botao para chamar o buscar funcionario↓

private void btnBuscarFuncionario\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaFuncionario = new List<string>();

ListaFuncionario.Add(txbEDCodFuncionario.Text);

ListaFuncionario.Add("");

ListaFuncionario.Add("");

ListaFuncionario.Add("");

ListaFuncionario.Add("");

ListaFuncionario.Add("");

ListaFuncionario.Add("");

ListaFuncionario.Add("");

Modelo.Controle controle = new Modelo.Controle();

Modelo.Funcionario funcionario = controle.PesquisarFuncionario(ListaFuncionario);

if (controle.mensagem.Equals(""))

{

txbEDCodFuncionario.Text = funcionario.Cod\_Funcionario;

txbEDNomeCompleto.Text = funcionario.Nome\_Completo;

txbEDNomeTratamento.Text = funcionario.Nome\_Tratamento;

txbEDCpf.Text = funcionario.CPF;

txbEDEndereco.Text = funcionario.End\_Completo;

txbEDTelefone.Text = funcionario.Telefone;

txbEDEmailContato.Text = funcionario.Email\_Contato;

}

else

{

MessageBox.Show(controle.mensagem);

}

}

//Código do botao para chamar o excluir funcionario↓

private void btnExcluirFuncionario\_Click\_1(object sender, RoutedEventArgs e)

{

String[] dados = { txbEDCodFuncionario.Text, txbEDNomeCompleto.Text, txbEDNomeTratamento.Text, txbEDCpf.Text, txbEDEndereco.Text, txbEDTelefone.Text, txbEDEmailContato.Text };

List<String> ListaFuncionario = new List<string>(dados);

Modelo.Controle controle = new Modelo.Controle();

MessageBoxResult resultado = MessageBox.Show("Deseja realmente excluir ?",

"Alerta de exclusão", MessageBoxButton.YesNo, MessageBoxImage.Warning);

if (resultado == MessageBoxResult.Yes)

{

controle.ExcluirFuncionario(ListaFuncionario);

MessageBox.Show(controle.mensagem);

}

}

//Código do botao para chamar o editar funcionario↓

private void btnEditarFuncionario\_Click\_1(object sender, RoutedEventArgs e)

{

List<String> ListaFuncionario = new List<string>();

ListaFuncionario.Add(txbEDCodFuncionario.Text);

ListaFuncionario.Add(txbEDNomeCompleto.Text);

ListaFuncionario.Add(txbEDNomeTratamento.Text);

ListaFuncionario.Add(txbEDCpf.Text);

ListaFuncionario.Add(txbEDEndereco.Text);

ListaFuncionario.Add(txbEDTelefone.Text);

ListaFuncionario.Add(txbEDEmailContato.Text);

string stat\_Funcionario = "";

if (rdbEDStatusFuncionarioAtivo.IsChecked == true) stat\_Funcionario = "A";

if (rdbEDStatusFuncionarioInativo.IsChecked == true) stat\_Funcionario = "I";

ListaFuncionario.Add(stat\_Funcionario);

ListaFuncionario.Add(txbEdIdUsuario.Text);

Modelo.Controle controle = new Modelo.Controle();

controle.EditarFuncionario(ListaFuncionario);

MessageBox.Show(controle.mensagem);

}

}

}

**frmPaginaPrincipal**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace SistemaEvolution.Apresentacao

{

//Código para iniciar o form principal↓

public partial class frmPaginaPrincipal : Window

{

public frmPaginaPrincipal()

{

InitializeComponent();

}

//Código para iniciar o form usuarios↓

private void mniPaginaUsuarios\_Click(object sender, RoutedEventArgs e)

{

Apresentacao.frmUsuario frmC = new Apresentacao.frmUsuario();

frmC.ShowDialog();

}

//Código para iniciar o form cliente↓

private void mniPaginaClientes\_Click(object sender, RoutedEventArgs e)

{

Apresentacao.frmClientes frmC = new Apresentacao.frmClientes();

frmC.ShowDialog();

}

//Código para iniciar o form produtos↓

private void mniPaginaProdutos\_Click(object sender, RoutedEventArgs e)

{

Apresentacao.frmProduto frmC = new Apresentacao.frmProduto();

frmC.ShowDialog();

}

//Código para iniciar o form chamados↓

private void mniPaginaChamados\_Click(object sender, RoutedEventArgs e)

{

Apresentacao.frmChamado frmC = new Apresentacao.frmChamado();

frmC.ShowDialog();

}

//Código para iniciar o form funcionários↓

private void mniPaginaFuncionario\_Click(object sender, RoutedEventArgs e)

{

Apresentacao.frmFuncionario frmC = new Apresentacao.frmFuncionario();

frmC.ShowDialog();

}

}

}

**frmProduto**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace SistemaEvolution.Apresentacao

{

//Código para iniciar o form produto↓

public partial class frmProduto : Window

{

public frmProduto()

{

InitializeComponent();

}

//Código do botão para cadastrar produto↓

private void btnCadastrarProduto\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaProduto = new List<string>();

ListaProduto.Add(txbCodProduto.Text);

ListaProduto.Add(txbDescProduto.Text);

ListaProduto.Add(txbCodCliente.Text);

Modelo.Controle controle = new Modelo.Controle();

controle.CadastrarProduto(ListaProduto);

MessageBox.Show(controle.mensagem);

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

}

//Código do botão para buscar o produto↓

private void btnBuscarProduto\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaProduto = new List<string>();

ListaProduto.Add(txbEdCodProduto.Text);

ListaProduto.Add("");

ListaProduto.Add("");

Modelo.Controle controle = new Modelo.Controle();

controle.PesquisarProduto(ListaProduto);

Modelo.Produto produto = new Modelo.Produto();

if (Modelo.atbEstaticos.listaProdutoEstatico == null)

{

MessageBox.Show("Nome Invalido");

}

else

if (Modelo.atbEstaticos.listaProdutoEstatico.Count() == 0)

{

MessageBox.Show("Não existe resposta para esta consulta");

}

else

if (Modelo.atbEstaticos.listaProdutoEstatico.Count() == 1)

{

produto = Modelo.atbEstaticos.listaProdutoEstatico[0];

txbEdCodProduto.Text = produto.Cod\_Produto;

txbEdDescricaoProduto.Text = produto.Desc\_Produto;

txbEDCodCliente.Text = produto.Cod\_Cliente;

}

}

//Código do botão para editar produto↓

private void btnEditarProduto\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaProduto = new List<string>();

ListaProduto.Add(txbEdCodProduto.Text);

ListaProduto.Add(txbEdDescricaoProduto.Text);

ListaProduto.Add(txbEDCodCliente.Text);

Modelo.Controle controle = new Modelo.Controle();

controle.EditarProduto(ListaProduto);

MessageBox.Show(controle.mensagem);

}

//Código do botão para excluir produto↓

private void btnExcluirProduto\_Click(object sender, RoutedEventArgs e)

{

String[] dados = { txbEdCodProduto.Text,txbEdDescricaoProduto.Text,txbEDCodCliente.Text };

List<String> ListaProduto= new List<string>(dados);

Modelo.Controle controle = new Modelo.Controle();

MessageBoxResult resultado = MessageBox.Show("Deseja realmente excluir ?",

"Alerta de exclusão", MessageBoxButton.YesNo, MessageBoxImage.Warning);

if (resultado == MessageBoxResult.Yes)

{

controle.ExcluirProduto(ListaProduto);

MessageBox.Show(controle.mensagem);

}

}

private void TextBox\_TextChanged(object sender, TextChangedEventArgs e)

{

}

}

}

**frmSelecao**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace SistemaEvolution.Apresentacao

{

//Código para inciar o form seleção↓

public partial class frmSelecao : Window

{

public frmSelecao()

{

InitializeComponent();

InicializarDTG();

}

//Código para inciar form DTG↓

public void InicializarDTG()

{

dtgSelecao.ItemsSource = Modelo.atbEstaticos.listaClienteEstatico;

}

//Código do para fechar o form dtg↓

private void btnConfirmarSelecao\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

//Código para selecionar a lista↓

private void dtgSelecao\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

Modelo.atbEstaticos.listaClienteEstatico.Clear();

Modelo.atbEstaticos.listaClienteEstatico.Add(

(Modelo.Cliente)dtgSelecao.SelectedItem);

}

}

}

**frmSelecaoFuncionario**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace SistemaEvolution.Apresentacao

{

//Código para iniciar o form↓

public partial class frmSelecaoFuncionario : Window

{

public frmSelecaoFuncionario()

{

InitializeComponent();

InicializarDTG();

}

//Código para iniciar o form dtg↓

public void InicializarDTG()

{

dtgSelecaoFuncionario.ItemsSource = Modelo.atbEstaticos.listaFuncionarioEstatico;

}

//Código para exibir o dtg↓

private void dtgSelecaoFuncionario\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

Modelo.atbEstaticos.listaFuncionarioEstatico.Clear();

Modelo.atbEstaticos.listaFuncionarioEstatico.Add(

(Modelo.Funcionario)dtgSelecaoFuncionario.SelectedItem);

}

//Código para fechar o from dtg↓

private void Button\_Click(object sender, RoutedEventArgs e)

{

this.Close();

}

}

}

**frmUsuario**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace SistemaEvolution.Apresentacao

{

//Código para iniciar o form↓

public partial class frmUsuario : Window

{

public frmUsuario()

{

InitializeComponent();

}

//Código do botao para chamar o cadastro usuario↓

private void btnCadastrarUsuario\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaUsuario = new List<string>();

ListaUsuario.Add(txbIdUsuario.Text);

ListaUsuario.Add(txbSenha.Text);

string Acesso = "";

if (rdbCliente.IsChecked == true) Acesso = "C";

if (rdbFuncionario.IsChecked == true) Acesso = "F";

ListaUsuario.Add(Acesso);

Modelo.EvolutionEntities acesso = new Modelo.EvolutionEntities();

Modelo.Controle controle = new Modelo.Controle();

controle.CadastrarUsuario(ListaUsuario);

MessageBox.Show(controle.mensagem);

}

//Código do botao para chamar o pesquisar usuario↓

private void btnPesquisarUsuario\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaUsuario = new List<string>();

ListaUsuario.Add(txbEDIdUsuario.Text);

ListaUsuario.Add("");

Modelo.Controle controle = new Modelo.Controle();

controle.PesquisarUsuario(ListaUsuario);

Modelo.Usuario usuario = new Modelo.Usuario();

if (Modelo.atbEstaticos.listaUsuarioEstatico == null)

{

MessageBox.Show("Nome Invalido");

}

else

if (Modelo.atbEstaticos.listaUsuarioEstatico.Count() == 0)

{

MessageBox.Show("Não existe resposta para esta consulta");

}

else

if (Modelo.atbEstaticos.listaUsuarioEstatico.Count() == 1)

{

usuario=Modelo.atbEstaticos.listaUsuarioEstatico[0];

txbEDIdUsuario.Text = usuario.ID\_usuario;

txbEDSenha.Text = usuario.Senha;

}

}

//Código do botao para chamar o excluir usuario↓

private void btnExcluirUsuario\_Click(object sender, RoutedEventArgs e)

{

String[] dados = { txbEDIdUsuario.Text, txbEDSenha.Text};

List<String> ListaUsuario = new List<string>(dados);

Modelo.Controle controle = new Modelo.Controle();

MessageBoxResult resultado = MessageBox.Show("Deseja realmente excluir ?",

"Alerta de exclusão", MessageBoxButton.YesNo, MessageBoxImage.Warning);

if (resultado == MessageBoxResult.Yes)

{

controle.ExcluirUsuario(ListaUsuario);

MessageBox.Show(controle.mensagem);

}

}

//Código do botao para chamar o editar usuario↓

private void btnEditarUsuario\_Click(object sender, RoutedEventArgs e)

{

List<String> ListaUsuario = new List<string>();

ListaUsuario.Add(txbEDIdUsuario.Text);

ListaUsuario.Add(txbEDSenha.Text);

Modelo.Controle controle = new Modelo.Controle();

controle.EditarUsuario(ListaUsuario);

MessageBox.Show(controle.mensagem);

}

}

}

**MainWindow**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using SistemaEvolution;

namespace SistemaEvolution

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

//Código para iniciar o form↓

public MainWindow()

{

InitializeComponent();

}

//Código do login do usuario↓

private void btnLogar\_Click(object sender, RoutedEventArgs e)

{

var login = new Modelo.Login();

string Email\_Contato;

string Senha;

Email\_Contato = txbLogin.Text;

Senha = txbSenha.Text;

if (login.Logon(Email\_Contato, Senha) == true)

{

Apresentacao.frmPaginaPrincipal frmC = new Apresentacao.frmPaginaPrincipal();

frmC.ShowDialog();

}

if (login.Logon(Email\_Contato, Senha) == false)

{

MessageBox.Show("Login ou Senha incorretos");

}

}

}

}

**Dao**

**Class - AtendimentoDAO**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using SistemaEvolution.Modelo;

namespace SistemaEvolution.DAL

{

public class AtendimentoDAO

{

//Declaração da variavel↓

EvolutionEntities TipoAtendimento = new EvolutionEntities();

public String mensagem;

//Código do cadastro do atendimento↓

public void CadastrarTipoAtendimento(Modelo.TipoAtendimento tipoAtendimento)

{

this.mensagem = "";

try

{

TipoAtendimento.TipoAtendimento.Add(tipoAtendimento);

TipoAtendimento.SaveChanges();

this.mensagem = "Atendimento cadastrado com sucesso";

}

catch (Exception e)

{

this.mensagem = "Código do Atendimento ja cadastrado, digite outro código.";

}

}

}

}

**Class - ChamadoDAO**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using SistemaEvolution.Modelo;

namespace SistemaEvolution.DAL

{

public class ChamadoDAO

{

//Declaração da variavel↓

EvolutionEntities Chamados = new EvolutionEntities();

public String mensagem;

//Código para pesquisar chamados↓

public List<Modelo.Chamados> PesquisarChamados(Modelo.Chamados chamados)

{

this.mensagem = "";

List<Modelo.Chamados> listaChamados = new List<Modelo.Chamados>();

IQueryable lista = from Chamados in Chamados.Chamados

where

Chamados.Cod\_Chamado.Contains(chamados.Cod\_Chamado)

select Chamados;

foreach (Modelo.Chamados p in lista)

{

listaChamados.Add(p);

}

return listaChamados;

}

}

}

**Class - ClienteDAO**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

using SistemaEvolution.Modelo;

namespace SistemaEvolution.DAL

{

public class ClienteDAO

{

//Declaraçao das variaveis ↓

EvolutionEntities Cliente = new EvolutionEntities();

public String mensagem;

//Codigo para Cadastrar o Cliente ↓

public void CadastrarCliente(Modelo.Cliente cliente)

{

try

{

Cliente.Cliente.Add(cliente);

Cliente.SaveChanges();

this.mensagem = "Cliente cadastrado com sucesso";

}

catch (Exception e)

{

this.mensagem = "Código do cliente ja cadastrado,digite outro código.";

}

}

//Codigo para Pesquisar o Cliente pelo ID ↓

public Modelo.Cliente PesquisarCliente(Modelo.Cliente cliente)

{

this.mensagem = "";

return Cliente.Cliente.Find(cliente.Cod\_Cliente);

}

//Codigo para pesquisar o Cliente pelo nome ↓

public List<Modelo.Cliente> PesquisarClientePorNome(Modelo.Cliente cliente)

{

this.mensagem = "";

List<Modelo.Cliente> listaCliente = new List<Modelo.Cliente>();

IQueryable lista = from Cliente in Cliente.Cliente

where

Cliente.Nome.Contains(cliente.Nome)

select Cliente;

foreach (Modelo.Cliente p in lista)

{

listaCliente.Add(p);

}

return listaCliente;

}

//Codigo para Excluir o Cliente ↓

public void ExcluirCliente(Modelo.Cliente cliente)

{

this.mensagem = "";

cliente = Cliente.Cliente.Find(cliente.Cod\_Cliente);

Cliente.Cliente.Remove(cliente);

Cliente.SaveChanges();

this.mensagem = "Pessoa excluída com sucesso !!!!!";

}

//Codigo para Editar o Cliente ↓

public void EditarCliente (Modelo.Cliente cliente)

{

this.mensagem = "";

Cliente.Entry(cliente).State = System.Data.EntityState.Modified;

Cliente.SaveChanges();

this.mensagem= "Pessoa editada com sucesso !!!!!";

}

}

}

**Class - ConexaoBD**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

namespace SistemaEvolution.DAL

{

//classe de conexão com banco↓

class Conexao

{

SqlConnection conexaoBD;

public Conexao()

{

conexaoBD = new SqlConnection();

conexaoBD.ConnectionString = @"Data Source=DESKTOP-6PI58CV\SQLEXPRESS;Initial Catalog=Evolution;Integrated Security=True";

}

public SqlConnection Conectar()

{

if (conexaoBD.State == System.Data.ConnectionState.Closed)

conexaoBD.Open();

return conexaoBD;

}

public void Desconectar()

{

if (conexaoBD.State == System.Data.ConnectionState.Open)

conexaoBD.Close();

}

}

}

**Class - FuncionarioDAO**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data.SqlClient;

using System.Threading.Tasks;

using SistemaEvolution.Modelo;

namespace SistemaEvolution.DAL

{

public class FuncionarioDAO

{

//Chamada do entity ↓

EvolutionEntities Funcionario = new EvolutionEntities();

public String mensagem;

//Código da comunicação do banco de dados para cadastrar o funcionário ↓

public void CadastrarFuncionario(Modelo.Funcionario funcionario)

{

try

{

Funcionario.Funcionario.Add(funcionario);

Funcionario.SaveChanges();

this.mensagem = "Funcionario cadastrado com sucesso";

}

catch (Exception e)

{

this.mensagem = "Código do funcionário ja cadastrado,digite outro código.";

}

}

//Código da comunicação do banco de dados para Pesquisar pelo id o funcionário ↓

public Modelo.Funcionario PesquisarFuncionario(Modelo.Funcionario funcionario)

{

this.mensagem = "";

return Funcionario.Funcionario.Find(funcionario.Cod\_Funcionario);

}

//Código da comunicação do banco de dados para Pesquisar pelo nome o funcionário ↓

public List<Modelo.Funcionario> PesquisarFuncionarioPorNome(Modelo.Funcionario funcionario)

{

this.mensagem = "";

List<Modelo.Funcionario> listaFuncionario = new List<Modelo.Funcionario>();

IQueryable lista = from Funcionario in Funcionario.Funcionario

where

Funcionario.Nome\_Completo.Contains(funcionario.Nome\_Completo)

select Funcionario;

foreach (Modelo.Funcionario p in lista)

{

listaFuncionario.Add(p);

}

return listaFuncionario; ;

}

//Código da comunicação do banco de dados para Excluir o funcionário ↓

public void ExcluirFuncionario(Modelo.Funcionario funcionario)

{

this.mensagem = "";

funcionario = Funcionario.Funcionario.Find(funcionario.Cod\_Funcionario);

Funcionario.Funcionario.Remove(funcionario);

Funcionario.SaveChanges();

this.mensagem = "Funcionário excluído com sucesso !!!!!";

}

//Código da comunicação do banco de dados para Editar o funcionário ↓

public void EditarFuncionario(Modelo.Funcionario funcionario)

{

this.mensagem = "";

Funcionario.Entry(funcionario).State = System.Data.EntityState.Modified;

Funcionario.SaveChanges();

this.mensagem = "Funcionário editado com sucesso !!!!!";

}

}

}

**Class - IntClienteDAO**

using System;

using SistemaEvolution.Modelo;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SistemaEvolution.DAL

{

interface intClienteDAO

{

void CadastrarCliente(Cliente cliente);

void EditarCliente(Cliente cliente);

void ExcluirCliente(Cliente cliente);

void PesquisarCliente(Cliente cliente);

void PesquisarClientePorNome(Cliente cliente);

}

}

**Class - IntFuncionarioDAO**

using System;

using SistemaEvolution.Modelo;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SistemaEvolution.DAL

{

interface intFuncionarioDAO

{

void CadastrarFuncionario(Funcionario funcionario);

void EditarFuncionario(Funcionario funcionario);

void ExcluirFuncionario(Funcionario funcionario);

void PesquisarFuncionario(Funcionario funcionario);

List<Cliente> PesquisarClientePorNome(Funcionario funcionario);

}

}

**Class - IntProdutoDAO**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SistemaEvolution.DAL

{

public class intProdutoDAO

{

}

}

**Class - IntUsuarioDAO**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SistemaEvolution.DAL

{

class intUsuarioDAO

{

}

}

**Class - ProdutoDAO**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using SistemaEvolution.Modelo;

namespace SistemaEvolution.DAL

{

public class ProdutoDAO

{

//Declaraçao das variaveis↓

EvolutionEntities Produto = new EvolutionEntities();

public String mensagem;

//Código para cadastrar produto↓

public void CadastrarProduto(Modelo.Produto produto)

{

this.mensagem = "";

try

{

Produto.Produto.Add(produto);

Produto.SaveChanges();

this.mensagem = "Produto cadastrado com sucesso";

}

catch (Exception e)

{

this.mensagem = "Código do produto ja cadastrado, digite outro código.";

}

}

//Codigo para Pesquisar o Produto pelo ID

public List<Modelo.Produto> PesquisarProduto(Modelo.Produto produto)

{

this.mensagem = "";

List<Modelo.Produto> listaProduto = new List<Modelo.Produto>();

IQueryable lista = from Produto in Produto.Produto

where

Produto.Cod\_Produto.Contains(produto.Cod\_Produto)

select Produto;

foreach (Modelo.Produto p in lista)

{

listaProduto.Add(p);

}

return listaProduto;

}

//Código para excluir produto↓

public void ExcluirProduto(Modelo.Produto produto)

{

this.mensagem = "";

produto = Produto.Produto.Find(produto.Cod\_Produto);

Produto.Produto.Remove(produto);

Produto.SaveChanges();

this.mensagem = "Produto excluída com sucesso !!!!!";

}

//Código para editar produto↓

public void EditarProduto(Modelo.Produto produto)

{

this.mensagem = "";

Produto.Entry(produto).State = System.Data.EntityState.Modified;

Produto.SaveChanges();

this.mensagem = "Produto editada com sucesso !!!!!";

}

}

}

**Class - UsuarioDAO**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

using SistemaEvolution.Modelo;

namespace SistemaEvolution.DAL

{

public class UsuarioDAO

{

//Declaraçao das variaveis↓

EvolutionEntities Usuario = new EvolutionEntities();

public String mensagem;

//Código para cadastrar o usuario↓

public void CadastrarUsuario(Modelo.Usuario usuario)

{

try

{

Usuario.Usuario.Add(usuario);

Usuario.SaveChanges();

this.mensagem = "Usuario cadastrado com sucesso";

}

catch (Exception e)

{

this.mensagem = "Código do cliente ja cadastrado,digite outro código.";

}

}

//Código para pesquisar o usuario pelo ID↓

public List<Modelo.Usuario> PesquisarUsuario(Modelo.Usuario usuario)

{

this.mensagem = "";

List<Modelo.Usuario> listaUsuario = new List<Modelo.Usuario>();

IQueryable lista = from Usuario in Usuario.Usuario

where

Usuario.ID\_usuario.Contains(usuario.ID\_usuario)

select Usuario;

foreach (Modelo.Usuario p in lista)

{

listaUsuario.Add(p);

}

return listaUsuario;

}

//Código para excluir o usuario↓

public void ExcluirUsuario(Modelo.Usuario usuario)

{

this.mensagem = "";

usuario = Usuario.Usuario.Find(usuario.ID\_usuario);

Usuario.Usuario.Remove(usuario);

Usuario.SaveChanges();

this.mensagem = "Usuário excluído com sucesso !!!!!";

}

//Código para editar o usuario↓

public void EditarUsuario(Modelo.Usuario usuario)

{

this.mensagem = "";

Usuario.Entry(usuario).State = System.Data.EntityState.Modified;

Usuario.SaveChanges();

this.mensagem = "Usuário editado com sucesso !!!!!";

}

}

}

**Modelo**

**Class - AtbEstaticos**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SistemaEvolution.Modelo

{

// Lista estatica de todas as classe ↓

public static class atbEstaticos

{

public static List<Cliente> listaClienteEstatico;

public static List<Funcionario> listaFuncionarioEstatico;

public static List<Produto> listaProdutoEstatico;

public static List<Usuario> listaUsuarioEstatico;

public static List<Chamados> listaChamadosEstatico;

}

}

**Class - Controle**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SistemaEvolution.Modelo

{

public class Controle

{

//Crud cliente ↓

//Código do Cadastrar cliente ↓

public String mensagem;

public void CadastrarCliente(List<String> ListaCliente)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosCliente(ListaCliente);

if (validacao.mensagem.Equals(""))

{

Cliente cliente = new Cliente();

cliente.Cod\_Cliente = ListaCliente[0];

cliente.Nome = ListaCliente[1];

cliente.Razao\_Social = ListaCliente[2];

cliente.CPF = ListaCliente[3];

cliente.CNPJ = ListaCliente[4];

cliente.Email\_Contato = ListaCliente[5];

cliente.End\_Completo = ListaCliente[6];

cliente.Telefone = ListaCliente[7];

cliente.Stat\_Cliente = ListaCliente[8];

cliente.ID\_usuario = ListaCliente[9];

DAL.ClienteDAO ClienteDAO = new DAL.ClienteDAO();

ClienteDAO.CadastrarCliente(cliente);

this.mensagem = ClienteDAO.mensagem;

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Código do Pesquisar pelo id cliente ↓

public Modelo.Cliente PesquisarCliente(List<String> ListaCliente)

{

this.mensagem = "";

Cliente cliente = new Cliente();

Validacao validacao = new Validacao();

validacao.ValidarDadosCliente(ListaCliente);

if (validacao.mensagem.Equals(""))

{

cliente.Cod\_Cliente = validacao.Cod\_Cliente;

DAL.ClienteDAO ClienteDAO = new DAL.ClienteDAO();

cliente = ClienteDAO.PesquisarCliente(cliente);

this.mensagem = ClienteDAO.mensagem;

}

else

{

this.mensagem = validacao.mensagem;

}

return cliente;

}

//Código do Pesquisar pelo nome cliente ↓

public void PesquisarClientePorNome(List<String> ListaCliente)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosCliente(ListaCliente);

if (validacao.mensagem.Equals(""))

{

DAL.ClienteDAO clienteDAO = new DAL.ClienteDAO();

Cliente cliente = new Cliente();

cliente.Nome = ListaCliente[1];

atbEstaticos.listaClienteEstatico = clienteDAO.PesquisarClientePorNome(cliente);

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Código do Excluir cliente ↓

public void ExcluirCliente(List<String> ListaCliente)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosCliente(ListaCliente);

if (validacao.mensagem.Equals(""))

{

Cliente cliente = new Cliente();

cliente.Cod\_Cliente = validacao.Cod\_Cliente;

DAL.ClienteDAO clienteDAO = new DAL.ClienteDAO();

if (clienteDAO.PesquisarCliente(cliente).Nome != null)

{

clienteDAO.ExcluirCliente(cliente);

this.mensagem = clienteDAO.mensagem;

}

else

{

this.mensagem = "Não existe este ID";

}

}

}

//Código do Editar cliente ↓

public void EditarCliente(List<String> ListaCliente)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosCliente(ListaCliente);

if (validacao.mensagem.Equals(""))

{

Cliente cliente = new Cliente();

cliente.Cod\_Cliente = validacao.Cod\_Cliente;

cliente.Cod\_Cliente = ListaCliente[0];

cliente.Nome = ListaCliente[1];

cliente.Razao\_Social = ListaCliente[2];

cliente.CPF = ListaCliente[3];

cliente.CNPJ = ListaCliente[4];

cliente.Email\_Contato = ListaCliente[5];

cliente.End\_Completo = ListaCliente[6];

cliente.Telefone = ListaCliente[7];

cliente.Stat\_Cliente = ListaCliente[8];

cliente.ID\_usuario = ListaCliente[9];

DAL.ClienteDAO clienteDAO = new DAL.ClienteDAO();

clienteDAO.EditarCliente(cliente);

this.mensagem = clienteDAO.mensagem;

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Crud Funcionario ↓

//Código do Cadastrar funcionário ↓

public void CadastrarFuncionario(List<String> ListaFuncionario)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosFuncionario(ListaFuncionario);

if (validacao.mensagem.Equals(""))

{

Funcionario funcionario = new Funcionario();

funcionario.Cod\_Funcionario = ListaFuncionario[0];

funcionario.Nome\_Completo = ListaFuncionario[1];

funcionario.Nome\_Tratamento = ListaFuncionario[2];

funcionario.CPF = ListaFuncionario[3];

funcionario.End\_Completo = ListaFuncionario[4];

funcionario.Telefone = ListaFuncionario[5];

funcionario.Email\_Contato = ListaFuncionario[6];

funcionario.Stat\_Funcionario = ListaFuncionario[7];

funcionario.ID\_usuario = ListaFuncionario[8];

DAL.FuncionarioDAO FuncionarioDAO = new DAL.FuncionarioDAO();

FuncionarioDAO.CadastrarFuncionario(funcionario);

this.mensagem = FuncionarioDAO.mensagem;

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Código do Pesquisar funcionário ↓

public Modelo.Funcionario PesquisarFuncionario(List<String> ListaFuncionario)

{

this.mensagem = "";

Funcionario funcionario = new Funcionario();

Validacao validacao = new Validacao();

validacao.ValidarDadosFuncionario(ListaFuncionario);

if (validacao.mensagem.Equals(""))

{

funcionario.Cod\_Funcionario = validacao.Cod\_Funcionario;

DAL.FuncionarioDAO FuncionarioDAO = new DAL.FuncionarioDAO();

funcionario = FuncionarioDAO.PesquisarFuncionario(funcionario);

this.mensagem = FuncionarioDAO.mensagem;

}

else

{

this.mensagem = validacao.mensagem;

}

return funcionario;

}

//Código do Pesquisar funcionário pelo nome ↓

public void PesquisarFuncionarioPorNome(List<String> ListaFuncionario)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosFuncionario(ListaFuncionario);

if (validacao.mensagem.Equals(""))

{

DAL.FuncionarioDAO FuncionarioDAO = new DAL.FuncionarioDAO();

Funcionario funcionario = new Funcionario();

funcionario.Nome\_Completo = ListaFuncionario[1];

atbEstaticos.listaFuncionarioEstatico = FuncionarioDAO.PesquisarFuncionarioPorNome(funcionario);

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Código do Excluir funcionário ↓

public void ExcluirFuncionario(List<String> ListaFuncionario)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosFuncionario(ListaFuncionario);

if (validacao.mensagem.Equals(""))

{

Funcionario funcionario = new Funcionario();

funcionario.Cod\_Funcionario = validacao.Cod\_Funcionario;

DAL.FuncionarioDAO funcionarioDAO = new DAL.FuncionarioDAO();

if (funcionarioDAO.PesquisarFuncionario(funcionario).Nome\_Completo != null)

{

funcionarioDAO.ExcluirFuncionario(funcionario);

this.mensagem = funcionarioDAO.mensagem;

}

else

{

this.mensagem = "Não existe este ID";

}

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Código do Editar funcionário ↓

public void EditarFuncionario(List<String> ListaFuncionario)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosFuncionario(ListaFuncionario);

if (validacao.mensagem.Equals(""))

{

Funcionario funcionario = new Funcionario();

funcionario.Cod\_Funcionario = validacao.Cod\_Cliente;

funcionario.Cod\_Funcionario = ListaFuncionario[0];

funcionario.Nome\_Completo = ListaFuncionario[1];

funcionario.Nome\_Tratamento = ListaFuncionario[2];

funcionario.CPF = ListaFuncionario[3];

funcionario.End\_Completo = ListaFuncionario[4];

funcionario.Telefone = ListaFuncionario[5];

funcionario.Email\_Contato = ListaFuncionario[6];

funcionario.Stat\_Funcionario = ListaFuncionario[7];

funcionario.ID\_usuario=ListaFuncionario[8];

DAL.FuncionarioDAO funcionarioDAO = new DAL.FuncionarioDAO();

funcionarioDAO.EditarFuncionario(funcionario);

this.mensagem = funcionarioDAO.mensagem;

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Crud Produto ↓

//Código do Cadastrar Produto ↓

public void CadastrarProduto(List<String> ListaProduto)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosProduto(ListaProduto);

if (validacao.mensagem.Equals(""))

{

Produto produto = new Produto();

produto.Cod\_Produto = ListaProduto[0];

produto.Desc\_Produto = ListaProduto[1];

produto.Cod\_Cliente = ListaProduto[2];

DAL.ProdutoDAO ProdutoDAO = new DAL.ProdutoDAO();

ProdutoDAO.CadastrarProduto(produto);

this.mensagem = ProdutoDAO.mensagem;

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Código do Pesquisar Produto ↓

public void PesquisarProduto(List<String> ListaProduto)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosProduto(ListaProduto);

if (validacao.mensagem.Equals(""))

{

DAL.ProdutoDAO ProdutoDAO = new DAL.ProdutoDAO();

Produto produto = new Produto();

produto.Cod\_Produto = ListaProduto[0];

produto.Cod\_Cliente = ListaProduto[2];

atbEstaticos.listaProdutoEstatico = ProdutoDAO.PesquisarProduto(produto);

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Código do Editar Produto ↓

public void EditarProduto(List<String> ListaProduto)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosProduto(ListaProduto);

if (validacao.mensagem.Equals(""))

{

Produto produto = new Produto();

produto.Cod\_Produto = validacao.Cod\_Produto;

produto.Cod\_Produto = ListaProduto[0];

produto.Desc\_Produto = ListaProduto[1];

produto.Cod\_Cliente = ListaProduto[2];

DAL.ProdutoDAO produtoDAO = new DAL.ProdutoDAO();

produtoDAO.EditarProduto(produto);

this.mensagem = produtoDAO.mensagem;

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Código do Excluir Produto ↓

public void ExcluirProduto(List<String> ListaProduto)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosProduto(ListaProduto);

if (validacao.mensagem.Equals(""))

{

Produto produto= new Produto();

produto.Cod\_Produto = validacao.Cod\_Produto;

DAL.ProdutoDAO produtoDAO = new DAL.ProdutoDAO();

if (produtoDAO.PesquisarProduto(produto) != null)

{

produtoDAO.ExcluirProduto(produto);

this.mensagem = produtoDAO.mensagem;

}

else

{

this.mensagem = "Não existe este ID";

}

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Crud Usuário ↓

//Código do Cadastrar Usuário ↓

public void CadastrarUsuario(List<String> ListaUsuario)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosUsuario(ListaUsuario);

if (validacao.mensagem.Equals(""))

{

Usuario usuario = new Usuario();

usuario.ID\_usuario = ListaUsuario[0];

usuario.Senha = ListaUsuario[1];

usuario.Acesso = ListaUsuario[2];

DAL.UsuarioDAO UsuarioDAO = new DAL.UsuarioDAO();

UsuarioDAO.CadastrarUsuario(usuario);

this.mensagem = UsuarioDAO.mensagem;

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Código do Pesquisar Usuário ↓

public void PesquisarUsuario(List<String> ListaUsuario)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosUsuario(ListaUsuario);

if (validacao.mensagem.Equals(""))

{

DAL.UsuarioDAO UsuarioDAO = new DAL.UsuarioDAO();

Usuario usuario = new Usuario();

usuario.ID\_usuario = ListaUsuario[0];

usuario.Senha = ListaUsuario[1];

atbEstaticos.listaUsuarioEstatico = UsuarioDAO.PesquisarUsuario(usuario);

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Código do Editar Usuário ↓

public void EditarUsuario(List<String> ListaUsuario)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosUsuario(ListaUsuario);

if (validacao.mensagem.Equals(""))

{

Usuario usuario = new Usuario();

usuario.ID\_usuario = validacao.ID\_usuario;

usuario.ID\_usuario = ListaUsuario[0];

usuario.Senha = ListaUsuario[1];

DAL.UsuarioDAO usuarioDAO = new DAL.UsuarioDAO();

usuarioDAO.EditarUsuario(usuario);

this.mensagem = usuarioDAO.mensagem;

}

else

{

this.mensagem = validacao.mensagem;

}

}

//Código do Excluir Usuário ↓

public void ExcluirUsuario(List<String> ListaUsuario)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosUsuario(ListaUsuario);

if (validacao.mensagem.Equals(""))

{

Usuario usuario = new Usuario();

usuario.ID\_usuario = validacao.ID\_usuario;

DAL.UsuarioDAO usuarioDAO = new DAL.UsuarioDAO();

if (usuarioDAO.PesquisarUsuario(usuario) != null)

{

usuarioDAO.ExcluirUsuario(usuario);

this.mensagem = usuarioDAO.mensagem;

}

else

{

this.mensagem = "Não existe este ID";

}

}

else

{

this.mensagem = validacao.mensagem;

}

}

public void CadastrarTipoAtendimento(List<String> ListaTipoAtendimento)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosAtendimento(ListaTipoAtendimento);

if (validacao.mensagem.Equals(""))

{

TipoAtendimento tipoAtendimento = new TipoAtendimento();

tipoAtendimento.Cod\_Atendimento = ListaTipoAtendimento[0];

tipoAtendimento.Descricao = ListaTipoAtendimento[1];

tipoAtendimento.Prioridade = ListaTipoAtendimento[2];

DAL.AtendimentoDAO atendimentoDAO = new DAL.AtendimentoDAO();

atendimentoDAO.CadastrarTipoAtendimento(tipoAtendimento);

this.mensagem = atendimentoDAO.mensagem;

}

else

{

this.mensagem = validacao.mensagem;

}

}

public void PesquisarChamados(List<String> ListaChamados)

{

this.mensagem = "";

Validacao validacao = new Validacao();

validacao.ValidarDadosChamados(ListaChamados);

if (validacao.mensagem.Equals(""))

{

DAL.ChamadoDAO ChamadosDAO = new DAL.ChamadoDAO();

Chamados chamados = new Chamados();

chamados.Cod\_Chamado = ListaChamados[0];

chamados.Desc\_Chamado = ListaChamados[1];

atbEstaticos.listaChamadosEstatico = ChamadosDAO.PesquisarChamados(chamados);

}

else

{

this.mensagem = validacao.mensagem;

}

}

}

}

**Class - Login**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using SistemaEvolution;

namespace SistemaEvolution.Modelo

{

public class Login

{

//Código do login do usuario↓

public bool Logon(String Email\_Contato, String Senha)

{

var db = new EvolutionEntities();

var senha = db.Usuario.FirstOrDefault(u => u.Senha == Senha);

var email = db.Funcionario.FirstOrDefault(u => u.Email\_Contato==Email\_Contato);

if (senha != null && email != null)

{

return true;

}

return false;

}

}

}

**Class - Validacao**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

using SistemaEvolution.Modelo;

using SistemaEvolution.Apresentacao;

namespace SistemaEvolution.Modelo

{

//Declaração das variaveis↓

public class Validacao

{

public String mensagem;

public String Cod\_Cliente;

public String Cod\_Produto;

public String Cod\_Funcionario;

public String ID\_usuario;

//Código de validação do cliente↓

public void ValidarDadosCliente(List<String> ListaCliente)

{

this.mensagem = "";

if (ListaCliente[0] == "")

this.mensagem = "Código do cliente está vazio \n";

if (ListaCliente[0].Length > 5)

this.mensagem = "Código com mais de 5 caracteres \n";

if (ListaCliente[1].Length > 50)

this.mensagem = "Nome com mais de 30 caracteres \n";

if (ListaCliente[1] == "")

this.mensagem = "Nome do cliente está vazio \n";

if (ListaCliente[2].Length > 50)

this.mensagem = "Razão Social com mais de 50 caracteres \n";

if (ListaCliente[3].Length > 11)

this.mensagem = "CPF com mais de 11 caracteres \n";

if (ListaCliente[4].Length > 12)

this.mensagem = "CNPJ com mais de 12 caracteres \n";

if (ListaCliente[5].Length > 50)

this.mensagem = "E-mail com mais de 50 caracteres \n";

if (ListaCliente[6].Length > 50)

this.mensagem = "Endereço com mais de 50 caracteres \n";

if (ListaCliente[7].Length > 11)

this.mensagem = "Telefone com mais de 11 caracteres \n";

try

{

this.Cod\_Cliente = (ListaCliente[0]);

}

catch (FormatException e)

{

this.mensagem += "ID inválido";

}

}

//Código de validação do produto↓

public void ValidarDadosProduto(List<String> ListaProduto)

{

this.mensagem = "";

if (ListaProduto[0] == "")

this.mensagem = "Código do produto está vazio \n";

if (ListaProduto[0].Length > 5)

this.mensagem = "Código do produto com mais de 5 caracteres \n";

if (ListaProduto[1].Length > 100)

this.mensagem = "Descriçao com mais de 100 caracteres \n";

if (ListaProduto[2].Length > 5)

this.mensagem = "Código do cliente com mais de 5 caracteres";

try

{

this.Cod\_Produto = (ListaProduto[0]);

}

catch (FormatException e)

{

this.mensagem += "ID inválido";

}

}

//Código de validação do funcionário↓

public void ValidarDadosFuncionario(List<String> ListaFuncionario)

{

this.mensagem = "";

if (ListaFuncionario[0] =="")

this.mensagem = "Código do funcionário está vazio \n";

if (ListaFuncionario[0].Length>5)

this.mensagem= "Código com mais de 5 caracteres \n";

if (ListaFuncionario[1] =="")

this.mensagem = "Nome do funcionário está vazio \n";

if (ListaFuncionario[1].Length > 50)

this.mensagem = "Nome com mais de 50 caracteres \n";

if (ListaFuncionario[2].Length > 50)

this.mensagem = "Nome com mais de 50 caracteres \n";

if (ListaFuncionario[3].Length > 11)

this.mensagem = "CPF com mais de 11 caracteres \n";

if (ListaFuncionario[4].Length > 50)

this.mensagem = "Endereço com mais de 50 caracteres \n";

if (ListaFuncionario[5].Length > 11)

this.mensagem = "Telefone com mais de 11 caracteres \n";

if (ListaFuncionario[6].Length > 50)

this.mensagem = "E-mail com mais de 50 caracteres \n";

try

{

this.Cod\_Funcionario = (ListaFuncionario[0]);

}

catch (FormatException e)

{

this.mensagem += "ID inválido";

}

}

//Código de validação do usuario↓

public void ValidarDadosUsuario(List<String> ListaUsuario)

{

this.mensagem = "";

if(ListaUsuario[0].Length>20)

this.mensagem = "Código com mais de 5 caracteres \n";

try

{

this.ID\_usuario = (ListaUsuario[0]);

}

catch (FormatException e)

{

this.mensagem += "ID inválido";

}

}

//Código de validação do chamado↓

public void ValidarDadosChamados(List<String> ListaChamados)

{

this.mensagem = "";

if (ListaChamados[0].Length > 8)

this.mensagem = "Código com mais de 8 caracteres \n";

try

{

this.Cod\_Produto = (ListaChamados[0]);

}

catch (FormatException e)

{

this.mensagem += "ID inválido";

}

}

//Código de validação do atendimento↓

public void ValidarDadosAtendimento(List<String> ListaTipoAtendimento)

{

this.mensagem = "";

if (ListaTipoAtendimento[0].Length > 8)

this.mensagem = "Código com mais de 5 caracteres \n";

if (ListaTipoAtendimento[0]=="")

this.mensagem = "Código está vazio \n";

if (ListaTipoAtendimento[2]=="")

this.mensagem = "Escolha uma prioridade \n";

try

{

this.Cod\_Produto = (ListaTipoAtendimento[0]);

}

catch (FormatException e)

{

this.mensagem += "ID inválido";

}

}

}

}

# ANEXO 2

**App\_Start**

**Class - BundleConfig**

using System.Web;

using System.Web.Optimization;

namespace HelpDesk

{

public class BundleConfig

{

// Para obter mais informações sobre o agrupamento, visite https://go.microsoft.com/fwlink/?LinkId=301862

public static void RegisterBundles(BundleCollection bundles)

{

bundles.Add(new ScriptBundle("~/bundles/jquery").Include(

"~/Scripts/jquery-{version}.js"));

// Use a versão em desenvolvimento do Modernizr para desenvolver e aprender com ela. Após isso, quando você estiver

// pronto para a produção, utilize a ferramenta de build em https://modernizr.com para escolher somente os testes que precisa.

bundles.Add(new ScriptBundle("~/bundles/modernizr").Include(

"~/Scripts/modernizr-\*"));

bundles.Add(new ScriptBundle("~/bundles/bootstrap").Include(

"~/Scripts/bootstrap.js"));

bundles.Add(new StyleBundle("~/Content/css").Include(

"~/Content/bootstrap.css",

"~/Content/site.css"));

}

}

}

**Class - FilterConfig**

using System.Web;

using System.Web.Mvc;

namespace HelpDesk

{

public class FilterConfig

{

public static void RegisterGlobalFilters(GlobalFilterCollection filters)

{

filters.Add(new HandleErrorAttribute());

}

}

}

**Class - RouteConfig**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using System.Web.Routing;

namespace HelpDesk

{

public class RouteConfig

{

public static void RegisterRoutes(RouteCollection routes)

{

routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");

routes.MapRoute(

name: "Default",

url: "{controller}/{action}/{id}",

defaults: new { controller = "Login", action = "Index", id = UrlParameter.Optional }

);

}

}

}

**Class - WebApiConfig**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web.Http;

namespace HelpDesk

{

public static class WebApiConfig

{

public static void Register(HttpConfiguration config)

{

// Serviços e configuração da API da Web

// Rotas da API da Web

config.MapHttpAttributeRoutes();

config.Routes.MapHttpRoute(

name: "DefaultApi",

routeTemplate: "api/{controller}/{id}",

defaults: new { id = RouteParameter.Optional }

);

}

}

}

**Controllers**

**Class - ChamadosController**

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web;

using System.Web.Http;

using System.Web.Mvc;

using HelpDesk.Models;

namespace HelpDesk.Controllers

{

public class ChamadosController : ApiController

{

public static Context db = new Context();

// GET api/Chamados

public IEnumerable<Chamados> Get()

{

return db.Chamados.ToList();

}

// GET api/Chamados/5

public Chamados Get(string id)

{

return db.Chamados.ToList().Where(chamado => chamado.Cod\_Chamado == id).First();

}

// POST api/Chamados

public HttpResponseMessage Post([FromBody]Chamados value)

{

if (value == null)

return Request.CreateResponse(HttpStatusCode.BadRequest);

var chamados = db.Chamados.ToList();

value.Cod\_Chamado = (chamados.Count() + 1).ToString();

value.Data = DateTime.Now;

db.Chamados.Add(value);

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

// PUT api/Chamados/5

public HttpResponseMessage Put(int id, [FromBody]Chamados value)

{

if (value == null)

return Request.CreateResponse(HttpStatusCode.BadRequest);

db.Chamados.Add(value);

db.Entry(value).State = System.Data.Entity.EntityState.Modified;

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

// DELETE api/Chamados/5

public HttpResponseMessage Delete(string id)

{

var chamado = db.Chamados.ToList().Where(c => c.Cod\_Chamado == id).First();

db.Chamados.Remove(chamado);

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

}

}

**Class - ClienteController**

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using HelpDesk.Models;

namespace HelpDesk.Controllers

{

public class ClienteController : ApiController

{

public static Context db = new Context();

// GET api/values

public IEnumerable<Cliente> Get()

{

return db.Cliente.ToList();

}

// GET api/values/5

public Cliente Get(string id)

{

return db.Cliente.ToList().Where(Cliente => Cliente.Cod\_Cliente == id).First();

}

// POST api/values

public HttpResponseMessage Post([FromBody]Cliente value)

{

if (value == null)

return Request.CreateResponse(HttpStatusCode.BadRequest);

db.Cliente.Add(value);

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

// PUT api/values/5

public HttpResponseMessage Put(int id, [FromBody]Cliente value)

{

if (value == null)

return Request.CreateResponse(HttpStatusCode.BadRequest);

db.Cliente.Add(value);

db.Entry(value).State = System.Data.Entity.EntityState.Modified;

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

// DELETE api/values/5

public HttpResponseMessage Delete(string id)

{

var chamado = db.Cliente.ToList().Where(c => c.Cod\_Cliente == id).First();

db.Cliente.Remove(chamado);

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

}

}

**Class - DashboardController**

using System.Linq;

using System.Web.Mvc;

using HelpDesk.Models;

namespace HelpDesk.Controllers

{

public class DashboardController : Controller

{

private static Context db = new Context();

public ActionResult Index()

{

string Cod\_usuario = (string)@Session["Login"];

ViewBag.Cod\_usuario = Cod\_usuario;

if (Cod\_usuario == null)

{

return View();

}

else

{

Usuario usuario;

Cliente cliente = null;

cliente = db.Cliente.ToList().Where(c => c.ID\_usuario == Cod\_usuario).FirstOrDefault();

if (cliente != null)

{

usuario = db.Usuario.ToList().Where(u => u.ID\_usuario == cliente.ID\_usuario).FirstOrDefault();

ViewBag.Chamados = new Chamados().GetListByUsuer(Cod\_usuario.ToString());

ViewBag.IsCliente = true;

}

else

{

Funcionario funcionario = null;

funcionario = db.Funcionario.ToList().Where(f => f.ID\_usuario == Cod\_usuario).FirstOrDefault();

if (funcionario != null)

{

usuario = db.Usuario.ToList().Where(u => u.ID\_usuario == funcionario.ID\_usuario).First();

ViewBag.Chamados = new Chamados().GetList();

ViewBag.IsCliente = false;

}

}

ViewBag.Title = "Dashboard";

ViewBag.Produtos = new Produto().GetProdutos(Cod\_usuario.ToString());

}

return View();

}

}

}

**Class - FuncionarioController**

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using HelpDesk.Models;

namespace HelpDesk.Controllers

{

public class FuncionarioController : ApiController

{

public static Context db = new Context();

// GET api/values

public IEnumerable<Funcionario> Get()

{

return db.Funcionario.ToList();

}

// GET api/values/5

public Funcionario Get(string id)

{

return db.Funcionario.ToList().Where(funcionario => funcionario.Cod\_Funcionario == id).First();

}

// POST api/values

public HttpResponseMessage Post([FromBody]Funcionario value)

{

if (value == null)

return Request.CreateResponse(HttpStatusCode.BadRequest);

db.Funcionario.Add(value);

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

// PUT api/values/5

public HttpResponseMessage Put(int id, [FromBody]Funcionario value)

{

if (value == null)

return Request.CreateResponse(HttpStatusCode.BadRequest);

db.Funcionario.Add(value);

db.Entry(value).State = System.Data.Entity.EntityState.Modified;

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

// DELETE api/values/5

public HttpResponseMessage Delete(string id)

{

var chamado = db.Funcionario.ToList().Where(c => c.Cod\_Funcionario == id).First();

db.Funcionario.Remove(chamado);

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

}

}

**Class - LoginController**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Web.Http;

using System.Web.Mvc;

using HelpDesk.Models;

namespace HelpDesk.Controllers

{

public class LoginController : Controller

{

public ActionResult Index([FromBody]string email, string senha)

{

ViewBag.Title = "Home Page";

if (email != null && senha != null)

{

var result = Usuario.Login(email, senha);

if (result != null)

{

Session["Login"] = result;

Response.Redirect("~/Dashboard/Index");

return null;

}

}

return View();

}

}

}

**Class - ProdutoController**

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using HelpDesk.Models;

namespace HelpDesk.Controllers

{

public class ProdutoController : ApiController

{

public static Context db = new Context();

// GET api/values

public IEnumerable<Produto> Get()

{

return db.Produto.ToList();

}

// GET api/values/5

public Produto Get(string id)

{

return db.Produto.ToList().Where(produto => produto.Cod\_Produto == id).First();

}

// POST api/values

public HttpResponseMessage Post([FromBody]Produto value)

{

if (value == null)

return Request.CreateResponse(HttpStatusCode.BadRequest);

db.Produto.Add(value);

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

// PUT api/values/5

public HttpResponseMessage Put(int id, [FromBody]Produto value)

{

if (value == null)

return Request.CreateResponse(HttpStatusCode.BadRequest);

db.Produto.Add(value);

db.Entry(value).State = System.Data.Entity.EntityState.Modified;

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

// DELETE api/values/5

public HttpResponseMessage Delete(string id)

{

var chamado = db.Produto.ToList().Where(c => c.Cod\_Produto == id).First();

db.Produto.Remove(chamado);

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

}

}

**Class - TipoAtendimentoController**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using HelpDesk.Models;

namespace HelpDesk.Controllers

{

public class TipoAtendimentoController : ApiController

{

public static Context db = new Context();

// GET api/values

public IEnumerable<TipoAtendimento> Get()

{

return db.TipoAtendimento.ToList();

}

// GET api/values/5

public TipoAtendimento Get(string id)

{

return db.TipoAtendimento.ToList().Where(tipoAtendimento => tipoAtendimento.Cod\_Atendimento == id).First();

}

// POST api/values

public TipoAtendimento Post([FromBody]TipoAtendimento value)

{

if (value == null)

return null;

var tipo = db.TipoAtendimento.ToList();

value.Cod\_Atendimento = (tipo.Count() + 1).ToString();

value.Data = DateTime.Now;

db.TipoAtendimento.Add(value);

db.SaveChanges();

return value;

}

// PUT api/values/5

public HttpResponseMessage Put(int id, [FromBody]TipoAtendimento value)

{

if (value == null)

return Request.CreateResponse(HttpStatusCode.BadRequest);

db.TipoAtendimento.Add(value);

db.Entry(value).State = System.Data.Entity.EntityState.Modified;

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

// DELETE api/values/5

public HttpResponseMessage Delete(string id)

{

var chamado = db.TipoAtendimento.ToList().Where(c => c.Cod\_Atendimento == id).First();

db.TipoAtendimento.Remove(chamado);

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

}

}

**Class - UploadFileController**

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web;

using System.Web.Http;

using HelpDesk.Models;

namespace HelpDesk.Controllers

{

public class UploadFileController : ApiController

{

public HttpResponseMessage Post()

{

var file = HttpContext.Current.Request.Files.Count > 0 ?

HttpContext.Current.Request.Files[0] : null;

if (file != null && file.ContentLength > 0)

{

var fileName = Path.GetFileName(file.FileName);

var path = Path.Combine(

HttpContext.Current.Server.MapPath("~/UploadedFiles"),

DateTime.Now.ToString("dd-MM-yyyy") + fileName

);

file.SaveAs(path);

return Request.CreateResponse(HttpStatusCode.OK);

}

return Request.CreateResponse(HttpStatusCode.InternalServerError);

}

}

}

**Class - UsuarioController**

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using HelpDesk.Models;

namespace HelpDesk.Controllers

{

public class UsuarioController : ApiController

{

public static Context db = new Context();

// GET api/values

public IEnumerable<Usuario> Get()

{

var t = db.Usuario.ToList(); ;

return t;

}

// GET api/values/5

public Usuario Get(string id)

{

return db.Usuario.ToList().Where(usuario => usuario.ID\_usuario == id).First();

}

// POST api/values

public HttpResponseMessage Post([FromBody]Usuario value)

{

if (value == null)

return Request.CreateResponse(HttpStatusCode.BadRequest);

db.Usuario.Add(value);

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

// PUT api/values/5

public HttpResponseMessage Put(int id, [FromBody]Usuario value)

{

if (value == null)

return Request.CreateResponse(HttpStatusCode.BadRequest);

db.Usuario.Add(value);

db.Entry(value).State = System.Data.Entity.EntityState.Modified;

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

// DELETE api/values/5

public HttpResponseMessage Delete(string id)

{

var usuario = db.Usuario.ToList().Where(u => u.ID\_usuario == id).First();

db.Usuario.Remove(usuario);

db.SaveChanges();

return Request.CreateResponse(HttpStatusCode.OK);

}

}

}

**Models**

**Class - Chamados**

namespace HelpDesk.Models

{

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

public partial class Chamados

{

[System.ComponentModel.DataAnnotations.Key]

public string Cod\_Chamado { get; set; }

public string Desc\_Chamado { get; set; }

public Nullable<System.DateTime> Data { get; set; }

public string Cod\_Produto { get; set; }

public string Cod\_Atendimento { get; set; }

public virtual TipoAtendimento TipoAtendimento { get; set; }

public virtual Produto Produto { get; set; }

private static Context db = new Context();

public List<Chamados> GetListByUsuer(string cod)

{

try

{

List<Chamados> list = new List<Chamados>();

List<Chamados> chamados = db.Chamados.ToList();

foreach (Chamados c in chamados)

{

c.Produto = db.Produto.ToList().Where(p => p.Cod\_Produto == c.Cod\_Produto).ToList<Produto>().FirstOrDefault();

c.TipoAtendimento = db.TipoAtendimento.ToList().Where(tipoAtendimento => tipoAtendimento.Cod\_Atendimento == c.Cod\_Atendimento).First();

if (c.Produto.Cod\_Cliente == cod)

list.Add(c);

}

return list.OrderByDescending(c => c.Data).ToList(); ;

}

catch

{

return null;

}

}

public List<Chamados> GetList()

{

List<Chamados> chamados = db.Chamados.ToList().OrderByDescending(c => c.Data).ToList();

return chamados;

}

}

}

**Class - Cliente**

namespace HelpDesk.Models

{

using System;

using System.Collections.Generic;

public partial class Cliente

{

[System.ComponentModel.DataAnnotations.Key]

public string Cod\_Cliente { get; set; }

public string Nome { get; set; }

public string Razao\_Social { get; set; }

public string CPF { get; set; }

public string CNPJ { get; set; }

public string Email\_Contato { get; set; }

public string End\_Completo { get; set; }

public string Telefone { get; set; }

public char Stat\_Cliente { get; set; }

public string ID\_usuario { get; set; }

public virtual Usuario Usuario { get; set; }

public virtual ICollection<Produto> Produto { get; set; }

}

}

**Class - Context**

namespace HelpDesk.Models

{

using System;

using System.Data.Entity;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Data.Entity.ModelConfiguration.Conventions;

/\* Classe context é responsavel por fazer as interaçoes entre os modelos e o banco de dados

\* sendo necessario apenas mapealos utilizando o objeto DbSete.

\* BdSete é umma classe que implementa o entity Framework

\*/

public class Context : DbContext

{

public DbSet<Chamados> Chamados { get; set; }

public DbSet<Cliente> Cliente { get; set; }

public DbSet<Funcionario> Funcionario { get; set; }

public DbSet<Produto> Produto { get; set; }

public DbSet<TipoAtendimento> TipoAtendimento { get; set; }

public DbSet<Usuario> Usuario { get; set; }

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

modelBuilder.Conventions.Remove<PluralizingTableNameConvention>();

base.OnModelCreating(modelBuilder);

}

public Context()

{

this.Configuration.ProxyCreationEnabled = false;

}

}

}

**Class - Funcionario**

namespace HelpDesk.Models

{

using System;

using System.Collections.Generic;

public partial class Funcionario

{

[System.ComponentModel.DataAnnotations.Key]

public string Cod\_Funcionario { get; set; }

public string Nome\_Completo { get; set; }

public string Nome\_Tratamento { get; set; }

public string CPF { get; set; }

public string End\_Completo { get; set; }

public string Telefone { get; set; }

public string Email\_Contato { get; set; }

public char Stat\_Funcionario { get; set; }

public string ID\_usuario { get; set; }

public virtual Usuario Usuario { get; set; }

}

}

**Class - Produto**

//------------------------------------------------------------------------------

// <auto-generated>

// This code was generated from a template.

//

// Manual changes to this file may cause unexpected behavior in your application.

// Manual changes to this file will be overwritten if the code is regenerated.

// </auto-generated>

//------------------------------------------------------------------------------

namespace HelpDesk.Models

{

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

public partial class Produto

{

[System.ComponentModel.DataAnnotations.Key]

public string Cod\_Produto { get; set; }

public string Desc\_Produto { get; set; }

public string Cod\_Cliente { get; set; }

private static Context db = new Context();

public List<Produto> GetProdutos(string codCliente)

{

return db.Produto.ToList().Where(p => p.Cod\_Cliente == codCliente).ToList<Produto>();

}

}

}

**Class - TipoAtendimento**

namespace HelpDesk.Models

{

using System;

using System.Collections.Generic;

public partial class TipoAtendimento

{

[System.ComponentModel.DataAnnotations.Key]

public string Cod\_Atendimento { get; set; }

public string Descricao { get; set; }

public Nullable<System.DateTime> Data { get; set; }

public string Prioridade { get; set; }

}

}

**Class - Usuario**

namespace HelpDesk.Models

{

using System;

using System.Collections.Generic;

using System.Linq;

public partial class Usuario

{

[System.ComponentModel.DataAnnotations.Key] // System informa que é a chave primaria, porque nao foi utiizada nomenclatura padrao no banco que seria ID

public string ID\_usuario { get; set; }

public char Acesso { get; set; }

public string Senha { get; set; }

private static Context db = new Context();

public static string Login(String email, String senha)

{

Usuario usuario = null;

try {

Cliente cliente = null;

cliente = db.Cliente.ToList().Where(c => c.Email\_Contato == email).FirstOrDefault();

if (cliente != null)

{

usuario = db.Usuario.ToList().Where(u => u.ID\_usuario == cliente.ID\_usuario).FirstOrDefault();

if (usuario.Senha == senha)

return usuario.ID\_usuario;

}

Funcionario funcionario = null;

funcionario = db.Funcionario.ToList().Where(f => f.Email\_Contato == email).FirstOrDefault();

if (funcionario != null)

{

usuario = db.Usuario.ToList().Where(u => u.ID\_usuario == funcionario.ID\_usuario).FirstOrDefault();

return usuario.ID\_usuario;

}

return null;

} catch (Exception ex)

{

return null;

}

}

}

}

# ANEXO 3

create database Evolution

go

use Evolution

go

CREATE TABLE Funcionario (

Cod\_Funcionario varchar(5) PRIMARY KEY,

Nome\_Completo varchar(30),

Nome\_Tratamento varchar(30),

CPF varchar(11),

End\_Completo varchar(50),

Email\_Contato varchar(30),

Telefone varchar(11),

Stat\_Funcionario char,

ID\_usuario varchar(8)

)

go

CREATE TABLE Cliente (

Cod\_Cliente varchar(5) PRIMARY KEY,

Nome varchar(50),

Razao\_Social varchar(30),

CPF varchar(11),

CNPJ varchar(12),

Email\_Contato varchar(30),

End\_Completo varchar(30),

Telefone varchar(11),

Stat\_Cliente char,

ID\_usuario varchar(8)

)

go

CREATE TABLE TipoAtendimento (

Cod\_Atendimento varchar(5) PRIMARY KEY,

Descricao varchar(300),

Data date,

Prioridade varchar(8)

)

go

CREATE TABLE Produto (

Cod\_Produto varchar(5) PRIMARY KEY,

Desc\_Produto varchar(100),

Cod\_Cliente varchar(5),

FOREIGN KEY(Cod\_Cliente) REFERENCES Cliente (Cod\_Cliente)

)

go

CREATE TABLE Usuario (

Acesso char,

ID\_usuario varchar(8) PRIMARY KEY,

Senha varchar(8)

)

go

CREATE TABLE Chamados (

Cod\_Chamado varchar(8) PRIMARY KEY,

Desc\_Chamado varchar(500),

Data date,

Cod\_Produto varchar(5),

Cod\_Atendimento varchar(5),

FOREIGN KEY(Cod\_Produto) REFERENCES Produto (Cod\_Produto),

FOREIGN KEY(Cod\_Atendimento) REFERENCES TipoAtendimento (Cod\_Atendimento)

)

go

ALTER TABLE Funcionario ADD FOREIGN KEY(ID\_usuario) REFERENCES Usuario (ID\_usuario)

ALTER TABLE Cliente ADD FOREIGN KEY(ID\_usuario) REFERENCES Usuario (ID\_usuario)

go

# ANEXO 4