**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";

}

}

}

}