Código do Sistema Ponto de Vendas

O sistema ponto de vendas, tem como finalidade implementar um pequeno sistema em WPF, que sirva de exemplo para a turma da disciplina “Ambiente de Programação e Banco de Dados”.

Este sistema deve realizar as seguintes operações: Cadastrar (Clientes e Produtos), Reajustar preço dos produtos (Todos e/ou individualmente), realizar vendas (só à vista) e emitir os seguintes relatórios (Relação de Clientes cadastrados, Catálogo de produtos e relatório de vendas realizadas por cliente.

O banco de dados para este sistema é o seguinte:

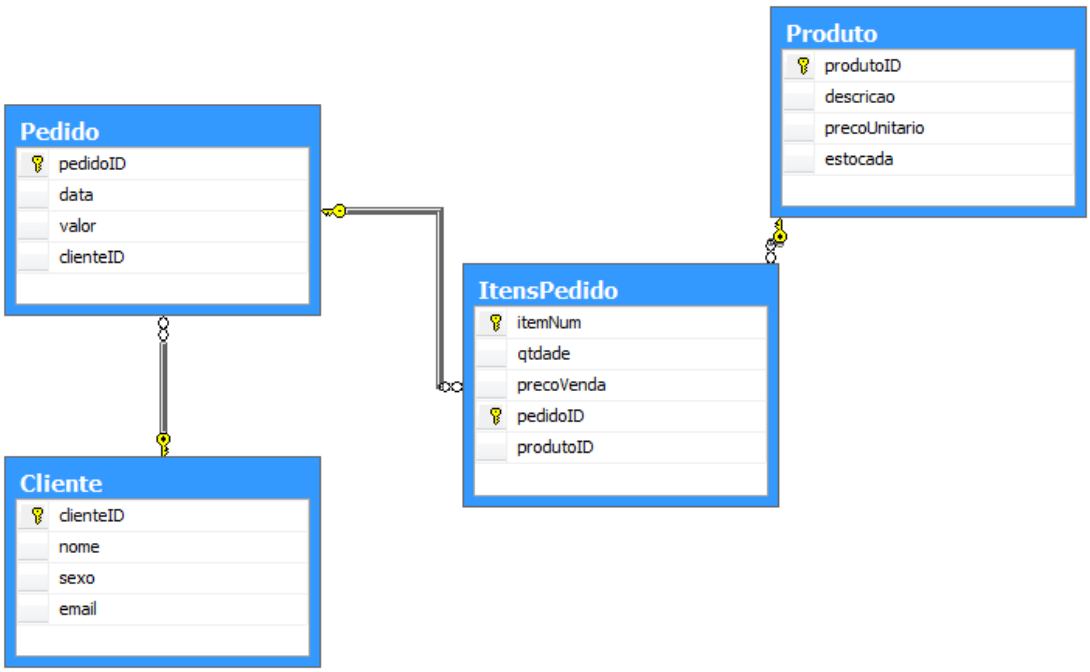

Figura 1: Banco de dados do Sistema Ponto de venda

Script para criar o banco de dados acima

```
Create Database PontoDeVenda

GO

USE [PontoVenda]
GO

CREATE TABLE [dbo].[Produto](
        [produtoID] [int] NOT NULL,
        [descricao] [varchar](50) NOT NULL,
        [precoUnitario] [decimal](12, 2) NOT NULL,
        [estocada] [int] NOT NULL,
```

```
 CONSTRAINT [PK__Produto__582517A17F60ED59] PRIMARY KEY CLUSTERED
(
	[produtoID] ASC
)
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[Cliente](
	[clienteID] [int] NOT NULL,
	[nome] [varchar](50) NOT NULL,
	[sexo] [char](1) NOT NULL,
	[email] [varchar](50) NULL,
 CONSTRAINT [PK__Cliente__C2FF24BD03317E3D] PRIMARY KEY CLUSTERED
(
	[clienteID] ASC
)
) ON [PRIMARY]
GO

CREATE PROCEDURE [dbo].[ReajustarPrecoDeUmProduto]
@produtoID int,
@percentual decimal (5,2)
AS
UPDATE Produto SET precoUnitario = precoUnitario * (1 + @percentual/100)
    WHERE produtoID=@produtoID
GO

CREATE PROCEDURE [dbo].[ReajustarPrecoDeProdutos]
@percentual decimal (5,2)
AS
UPDATE Produto SET precoUnitario = precoUnitario * (1 + @percentual/100)
GO

CREATE PROCEDURE [dbo].[UpdateProduto]
@produtoID int,
@descricao varchar(50),
@precoUnitario decimal (12,2),
@estocada int
AS
  UPDATE Produto SET produtoID = @produtoID,
		descricao = @descricao,
		precoUnitario = @precoUnitario,
		estocada = @estocada
		WHERE produtoID = @produtoID
GO

CREATE PROCEDURE [dbo].[AtualizarEstoque]
@produtoID int,
@qtdade int
AS
UPDATE Produto Set estocada = estocada - @qtdade WHERE produtoID = @produtoID
GO

CREATE PROCEDURE [dbo].[AtualizarCliente]
  @clienteid int,
```

```
  @nome varchar(50),
  @sexo char(01),
  @email varchar(50)
  AS
  UPDATE Cliente SET clienteID = @clienteID,
                nome      = @nome,
                sexo      = @sexo,
                email     = @email
           where clienteID = @clienteid
GO

CREATE PROCEDURE [dbo].[DeleteProduto]
@produtoID int
AS
 DELETE FROM Produto WHERE produtoID = @produtoID
GO

CREATE PROCEDURE [dbo].[DeletarCliente]
 @clienteID int
 AS
 DELETE FROM Cliente Where clienteID = @clienteID
GO

CREATE PROCEDURE [dbo].[InsertProduto]
@produtoID int,
@descricao varchar(50),
@precoUnitario decimal (12,2),
@estocada int
AS
 INSERT INTO Produto VALUES (@produtoID,@descricao,@precoUnitario,@estocada)
GO

CREATE PROCEDURE [dbo].[InserirCliente]
  @clienteid int,
  @nome varchar(50),
  @sexo char(01),
  @email varchar(50)
  AS
  INSERT INTO Cliente (clienteID,nome,sexo,email)
          values (@clienteid,@nome,@sexo,@email)
GO

CREATE PROCEDURE [dbo].[GetProdutos]
AS
 SELECT * FROM Produto
GO

CREATE PROCEDURE [dbo].[GetProduto]
@produtoID int
AS
 SELECT * FROM Produto WHERE produtoID = @produtoID
GO

CREATE TABLE [dbo].[Pedido](
        [pedidoID] [int] NOT NULL,
```

```
	[data] [date] NOT NULL,
	[valor] [decimal](12, 2) NULL,
	[clienteID] [int] NULL,
 CONSTRAINT [PK__Pedido__BAF07AE40F975522] PRIMARY KEY CLUSTERED
(
	[pedidoID] ASC
)
) ON [PRIMARY]
GO

CREATE PROCEDURE [dbo].[LocalizarTodosClientes]
  AS
  Select * from Cliente
GO

CREATE PROCEDURE [dbo].[LocalizarClientePorID]
  @clienteID int
  AS
  SELECT * FROM Cliente Where clienteID = @clienteID
GO

CREATE TABLE [dbo].[ItensPedido](
	[itemNum] [int] NOT NULL,
	[qtdade] [int] NOT NULL,
	[precoVenda] [decimal](12, 2) NOT NULL,
	[pedidoID] [int] NOT NULL,
	[produtoID] [int] NOT NULL,
 CONSTRAINT [pk_ItensPedido] PRIMARY KEY CLUSTERED
(
	[itemNum] ASC,
	[pedidoID] ASC
)
) ON [PRIMARY]
GO

CREATE PROCEDURE [dbo].[GetPedidos]
@clienteID int
AS
SELECT * FROM Pedido WHERE clienteID = @clienteID
GO

CREATE PROCEDURE [dbo].[InsertPedido]
@pedidoID int,
@data date,
@valor decimal(12,2),
@clienteID int
AS
INSERT INTO Pedido VALUES (@pedidoID,@data,@valor,@clienteID)
GO

CREATE PROCEDURE [dbo].[UtimoPedido]
AS

 if (Select MAX(PedidoID) FROM Pedido) = null
  return Convert(int, 0)
```

```
else
  Select CONVERT(int, MAX(PedidoID)) FROM Pedido
GO

CREATE PROCEDURE [dbo].[RealizarVendas]
@pedidoID int,
@clienteID int,
@itemNum int,
@qtdade int,
@produtoID int,
@precoVenda decimal (12,2)
AS
BEGIN
  BEGIN TRANSACTION
  BEGIN TRY
    INSERT INTO ItensPedido VALUES
(@itemNum,@qtdade,@precoVenda,@pedidoID,@produtoID)
    UPDATE Produto SET estocada=estocada - @qtdade WHERE produtoID=@produtoID
    COMMIT TRANSACTION
  END TRY
  BEGIN CATCH
     ROLLBACK TRANSACTION
  END CATCH
END
GO

CREATE PROCEDURE [dbo].[InsertItensPedido]
@itemNum int,
@qtade int,
@precoVenda decimal(12,2),
@pedidoID int,
@produtoID int
AS
INSERT INTO ItensPedido VALUES
(@itemNum,@qtade,@precoVenda,@pedidoID,@produtoID)
GO

CREATE PROCEDURE [dbo].[GetItensPedidoPorPedidoID]
@pedidoID int
AS
SELECT * FROM ItensPedido WHERE pedidoID = @pedidoID
GO

CREATE PROCEDURE [dbo].[GetItensPedido]
AS
SELECT * FROM ItensPedido
GO

ALTER TABLE [dbo].[Pedido] CHECK CONSTRAINT [fk_CliPed]
GO

ALTER TABLE [dbo].[ItensPedido]  WITH CHECK ADD  CONSTRAINT [fk_ItemProduto]
FOREIGN KEY([produtoID])
REFERENCES [dbo].[Produto] ([produtoID])
GO
```

```
ALTER TABLE [dbo].[ItensPedido] CHECK CONSTRAINT [fk_ItemProduto]
GO

ALTER TABLE [dbo].[ItensPedido]  WITH CHECK ADD  CONSTRAINT [fk_PediItem]
FOREIGN KEY([pedidoID])
REFERENCES [dbo].[Pedido] ([pedidoID])
GO
ALTER TABLE [dbo].[ItensPedido] CHECK CONSTRAINT [fk_PediItem]
GO
```

Diagrama de classes do sistema Ponto de Vendas

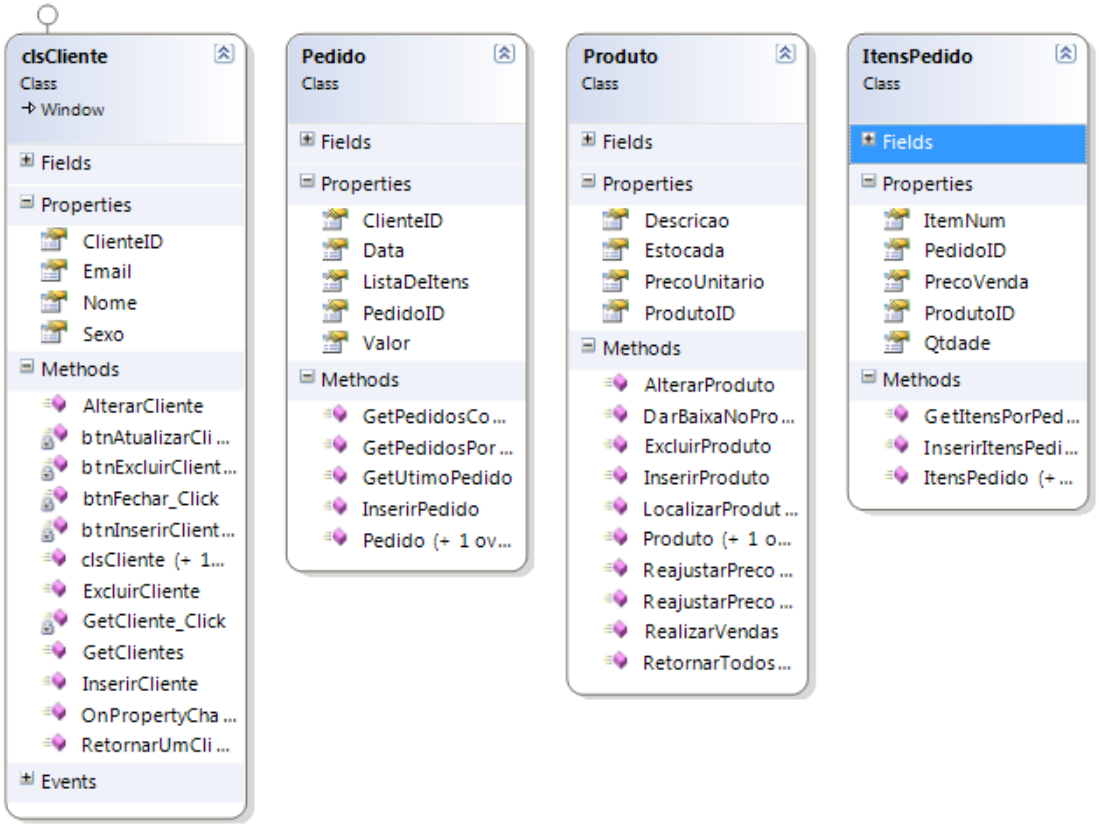

Figura 2: Diagrama de classes do sistema Ponto de vendas

C# - implementação do classe Cliente

Antes de implementar a classe clsCliente, adicione o arquivo App.config, abaixo:

App.config

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <appSettings>
```

```
   <add key="conString" value="Data Source=ServidorSql;Initial Catalog=PontoVenda;User
Id=senha; password=sa"/>
  </appSettings>
</configuration>
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data.SqlClient;
using System.Configuration;
using System.Data;
using System.Windows;
using System.ComponentModel;

namespace SistemaPontoVenda
{
   public partial class clsCliente
   {
      private SqlConnection con =null;

      private int clienteID;
      public int ClienteID
      {
         get { return clienteID; }
         set { clienteID = value;}
      }
      private string nome;

      public string Nome
      {
         get { return nome; }
         set { nome = value; }
      }
      private string sexo;

      public string Sexo
      {
         get { return sexo; }
         set { sexo = value; }
      }
      private string email;

      public string Email
      {
         get { return email; }
         set { email = value;}
      }

      public clsCliente(int clienteID, string nome, string sexo, string email)
      {
         this.ClienteID = clienteID;
         this.Nome = nome;
         this.Sexo = sexo;
         this.Email = email;
```

```
}


public void InserirCliente(clsCliente cliente)
{
   con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

   try
   {
      con.Open();
      SqlCommand cmd = new SqlCommand("InserirCliente", con);
      cmd.Connection = con;
      cmd.CommandType = System.Data.CommandType.StoredProcedure;
      cmd.Parameters.AddWithValue("@clienteID", cliente.ClienteID);
      cmd.Parameters.AddWithValue("@nome", cliente.Nome);
      cmd.Parameters.AddWithValue("@sexo", cliente.Sexo);
      cmd.Parameters.AddWithValue("@email", cliente.Email);

      cmd.ExecuteNonQuery();
   }
   catch (SqlException ex)
   {
      throw new Exception("Falha na operação." + ex.Message);
   }
}
public void AlterarCliente(clsCliente cliente)
{
   con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

   try
   {
      con.Open();
      SqlCommand cmd = new SqlCommand("AtualizarCliente", con);
      cmd.Connection = con;
      cmd.CommandType = System.Data.CommandType.StoredProcedure;
      cmd.Parameters.AddWithValue("@clienteID", cliente.ClienteID);
      cmd.Parameters.AddWithValue("@nome", cliente.Nome);
      cmd.Parameters.AddWithValue("@sexo", cliente.Sexo);
      cmd.Parameters.AddWithValue("@email", cliente.Email);

      cmd.ExecuteNonQuery();
   }
   catch (SqlException ex)
   {
      throw new Exception("Falha na operação." + ex.Message);
   }
}
public void ExcluirCliente(clsCliente cliente)
{
   con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

   try
   {
      con.Open();
      SqlCommand cmd = new SqlCommand("DeletarCliente", con);
```

```
            cmd.Connection = con;
            cmd.CommandType = System.Data.CommandType.StoredProcedure;
            cmd.Parameters.AddWithValue("@clienteID", cliente.ClienteID);

            cmd.ExecuteNonQuery();
        }
        catch (SqlException ex)
        {
            throw new Exception("Falha na operação." + ex.Message);
        }
    }
    public clsCliente RetornarUmCliente(clsCliente cliente)
    {
        con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);
        SqlDataReader reader = null;

        try
        {
            con.Open();
            SqlCommand cmd = new SqlCommand("LocalizarClientePorID", con);
            cmd.Connection = con;
            cmd.CommandType = System.Data.CommandType.StoredProcedure;
            cmd.Parameters.AddWithValue("@clienteID", cliente.ClienteID);

            reader = cmd.ExecuteReader(CommandBehavior.CloseConnection);
            clsCliente rcliente = new clsCliente();

            while (reader.Read())
            {
                rcliente.ClienteID = Convert.ToInt16(reader["clienteID"]);
                rcliente.Nome = reader["nome"].ToString();
                rcliente.Sexo = reader["sexo"].ToString();
                rcliente.Email = reader["email"].ToString();
            }
            return rcliente;
        }
        catch (SqlException ex)
        {
            throw new Exception("Falha na operação." + ex.Message);
        }
        finally
        {
            reader.Close();
            con.Close();
        }
    }

    public List<clsCliente> GetClientes()
    {
        con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);
        SqlDataReader reader = null;
        try
        {
            con.Open();
            SqlCommand cmd = new SqlCommand("LocalizarTodosClientes", con);
```

```
            cmd.CommandType = CommandType.StoredProcedure;

            reader = cmd.ExecuteReader(CommandBehavior.CloseConnection);

            //Lista de todos os produtos retornados
            List<clsCliente> listaClientes = new List<clsCliente>();

            //Lê o registro reornado, se foi localizado
            while (reader.Read())
            {
               clsCliente rcliente = new clsCliente();
               rcliente.ClienteID = Convert.ToInt16(reader["clienteID"]);
               rcliente.Nome = reader["nome"].ToString();
               rcliente.Sexo = reader["sexo"].ToString();
               rcliente.Email = reader["email"].ToString();
               listaClientes.Add(rcliente);
            }
            return listaClientes;
         }
         catch (Exception ex)
         {
            throw new Exception("Falha na operação: " + ex.Message);
         }
         finally
         {
            reader.Close();
            con.Close();
         }
      }
   }
}
```

C# - implementação da classe Produto – não esqueça de acrescentar o namespace System.Configuration

---

```
Using System.Configuration;

namespace SistemaPontoVenda
{
   public class Produto
   {
      private SqlConnection con = null;
      private SqlCommand cmd = null;

      private int produtoID;

      public int ProdutoID
      {
         get { return produtoID; }
         set { produtoID = value; }
      }
```

```
private string descricao;

public string Descricao
{
   get { return descricao; }
   set { descricao = value; }
}
private decimal precoUnitario;

public decimal PrecoUnitario
{
   get { return precoUnitario; }
   set { precoUnitario = value; }
}
private int estocada;

public int Estocada
{
   get { return estocada; }
   set { estocada = value; }
}

public Produto() { } //Construtor padrão

public Produto(int produtoID, string descricao, decimal precoUnitario, int estocada)
{
   this.ProdutoID = produtoID;
   this.Descricao = descricao;
   this.PrecoUnitario = precoUnitario;
   this.Estocada = estocada;
}


public void InserirProduto(Produto produto)
{
   con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

   try
   {
      con.Open();
      cmd = new SqlCommand("InsertProduto", con);
      cmd.CommandType = CommandType.StoredProcedure;
      cmd.Parameters.AddWithValue("@produtoID", produto.ProdutoID);
      cmd.Parameters.AddWithValue("@descricao", produto.Descricao);
      cmd.Parameters.AddWithValue("@precoUnitario", produto.PrecoUnitario);
      cmd.Parameters.AddWithValue("@estocada", produto.Estocada);

      //Executa o comando para realizar o cadastro no banco de dado
      cmd.ExecuteNonQuery();
   }
   catch (Exception ex)
   {
      throw new Exception("Falha na operação: " + ex.Message);
   }
   finally
```

```
    {
        con.Close();
    }
}

public void AlterarProduto(Produto produto)
{
    con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

    try
    {
        con.Open();
        cmd = new SqlCommand("UpdateProduto", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.AddWithValue("@produtoID", produto.ProdutoID);
        cmd.Parameters.AddWithValue("@descricao", produto.Descricao);
        cmd.Parameters.AddWithValue("@precoUnitario", produto.PrecoUnitario);
        cmd.Parameters.AddWithValue("@estocada", produto.Estocada);

        //Executa o comando para realizar a alteração do produto no banco de dado
        cmd.ExecuteNonQuery();
    }
    catch (Exception ex)
    {
        throw new Exception("Falha na operação: " + ex.Message);
    }
    finally
    {
        con.Close();
    }
}
public void ExcluirProduto(Produto produto)
{
    con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

    try
    {
        con.Open();
        cmd = new SqlCommand("DeleteProduto", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.AddWithValue("@produtoID", produto.ProdutoID);

        //Executa o comando para realizar a exclusão do produto no banco de dado
        cmd.ExecuteNonQuery();
    }
    catch (Exception ex)
    {
        throw new Exception("Falha na operação: " + ex.Message);
    }
    finally
    {
        con.Close();
    }
}
public Produto LocalizarProdutoPorID(Produto produto)
```

```
{
    con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);
    SqlDataReader reader = null;
    try
    {
        con.Open();
        cmd = new SqlCommand("GetProduto", con);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.AddWithValue("@produtoID", produto.ProdutoID);
        reader = cmd.ExecuteReader(CommandBehavior.CloseConnection);

        //Produto a ser retornado
        Produto rproduto = null;
        //Lê o registro reornado, se foi localizado
        while (reader.Read())
        {
            rproduto = new Produto();
            rproduto.ProdutoID = Convert.ToInt16(reader["produtoID"]);
            rproduto.Descricao = reader["descricao"].ToString();
            rproduto.PrecoUnitario = Convert.ToDecimal(reader["precoUnitario"]);
            rproduto.Estocada = Convert.ToInt16(reader["estocada"]);
        }
        return rproduto;
    }
    catch (Exception ex)
    {
        throw new Exception("Falha na operação: " + ex.Message);
    }
    finally
    {
        reader.Close();
        con.Close();
    }
}
public List<Produto> RetornarTodosProdutos()
{
    con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);
    SqlDataReader reader = null;
    try
    {
        con.Open();
        cmd = new SqlCommand("GetProdutos", con);
        cmd.CommandType = CommandType.StoredProcedure;

        reader = cmd.ExecuteReader(CommandBehavior.CloseConnection);

        //Lista de todos os produtos retornados
        List<Produto> listaProdutos = new List<Produto>();

        //Lê o registro reornado, se foi localizado
        while (reader.Read())
        {
            Produto rproduto = new Produto();
            rproduto.ProdutoID = Convert.ToInt16(reader["produtoID"]);
            rproduto.Descricao = reader["descricao"].ToString();
```

```
                rproduto.PrecoUnitario = Convert.ToDecimal(reader["precoUnitario"]);
                rproduto.Estocada = Convert.ToInt16(reader["estocada"]);
                listaProdutos.Add(rproduto);
            }
            return listaProdutos;
        }
        catch (Exception ex)
        {
            throw new Exception("Falha na operação: " + ex.Message);
        }
        finally
        {
            reader.Close();
            con.Close();
        }
    }
    public void RealizarVendas(int pedidoID, DateTime data, decimal valor,int clienteID, int
itemNum,
        int qtdade, int produtoID, decimal precoVenda)
    {
        con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

        try
        {
            con.Open();
            cmd = new SqlCommand("RealizarVendas", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.AddWithValue("@pedidoID", pedidoID);
            cmd.Parameters.AddWithValue("@data", data);
            cmd.Parameters.AddWithValue("@valor", valor);
            cmd.Parameters.AddWithValue("@clienteID", clienteID);
            cmd.Parameters.AddWithValue("@itemNum", itemNum);
            cmd.Parameters.AddWithValue("@qtdade", qtdade);
            cmd.Parameters.AddWithValue("produtoID", produtoID);
            cmd.Parameters.AddWithValue("precoVenda", precoVenda);

            //Executa o comando para realizar a alteração do produto no banco de dado
            cmd.ExecuteNonQuery();
        }
        catch (Exception ex)
        {
            throw new Exception("Falha na operação: " + ex.Message);
        }
        finally
        {
            con.Close();
        }
    }

    public void DarBaixaNoProduto(int produtoID, int qtdade)
    {
        con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

        try
        {
```

```
            con.Open();
            cmd = new SqlCommand("AtualizarEstoque", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.AddWithValue("produtoID", produtoID);
            cmd.Parameters.AddWithValue("@qtdade", qtdade);

            //Executa o comando para realizar a baixa na qtdade estocada do produto no banco de
dado
            cmd.ExecuteNonQuery();
        }
        catch (Exception ex)
        {
            throw new Exception("Falha na operação: " + ex.Message);
        }
        finally
        {
            con.Close();
        }
    }
    public void ReajustarPrecoDeUmProduto(int produtoID, decimal percentual)
    {
        con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

        try
        {
            con.Open();
            cmd = new SqlCommand("ReajustarPrecoDeUmProduto", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.AddWithValue("produtoID", produtoID);
            cmd.Parameters.AddWithValue("@percentual", percentual);

            //Executa o comando para realizar o reajuste de preço do produto no banco de dado
            cmd.ExecuteNonQuery();
        }
        catch (Exception ex)
        {
            throw new Exception("Falha na operação: " + ex.Message);
        }
        finally
        {
            con.Close();
        }
    }
    public void ReajustarPrecoDosProdutos(decimal percentual)
    {
        con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

        try
        {
            con.Open();
            cmd = new SqlCommand("ReajustarPrecoDeProdutos", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.AddWithValue("@percentual", percentual);

            //Executa o comando para realizar o reajuste de preço de todos os produto.
```

```
            cmd.ExecuteNonQuery();
        }
        catch (Exception ex)
        {
            throw new Exception("Falha na operação: " + ex.Message);
        }
        finally
        {
            con.Close();
        }
    }
  }
}
```

C# - implementação da classe Pedido

```
namespace SistemaPontoVenda
{
  public class Pedido
  {
    SqlConnection con = null;
    SqlCommand cmd = null;

    private int pedidoID;

    public int PedidoID
    {
      get { return pedidoID; }
      set { pedidoID = value; }
    }
    private DateTime data;

    public DateTime Data
    {
      get { return data; }
      set { data = value; }
    }
    private decimal valor;

    public decimal Valor
    {
      get { return valor; }
      set { valor = value; }
    }
    private int clienteID;

    public int ClienteID
    {
      get { return clienteID; }
      set { clienteID = value; }
    }
```

```
private List<ItensPedido> listaDeItens;

public List<ItensPedido> ListaDeItens
{
   get { return listaDeItens; }
   set
   {
      listaDeItens = value;
   }
}
public Pedido (int pedidoID, List<ItensPedido> listaDeItens)
{
   this.PedidoID = pedidoID;
   this.ListaDeItens = listaDeItens;
}

public Pedido() { }

public void InserirPedido(Pedido pedido)
{
   con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

   try
   {
      con.Open();
      cmd = new SqlCommand("InsertPedido", con);
      cmd.CommandType = CommandType.StoredProcedure;
      cmd.Parameters.AddWithValue("@pedidoID", pedido.PedidoID );
      cmd.Parameters.AddWithValue("@data", pedido.Data );
      cmd.Parameters.AddWithValue("@valor", pedido.Valor);
      cmd.Parameters.AddWithValue("@clienteID", pedido.ClienteID);

      //Executa o comando para inserir o pedido no banco de dado
      cmd.ExecuteNonQuery();
   }
   catch (Exception ex)
   {
      throw new Exception("Falha na operação: " + ex.Message);
   }
   finally
   {
      con.Close();
   }
}
public List<Pedido> GetPedidosPorClienteID(int clienteID)
{
   con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

   try
   {
      con.Open();
      cmd = new SqlCommand("GetPedidos", con);
      cmd.CommandType = CommandType.StoredProcedure;
      cmd.Parameters.AddWithValue("@clienteID", clienteID);
```

```
            SqlDataReader reader = cmd.ExecuteReader();
            //Cria uma lista de pedidos para recebe
            List<Pedido> listaPedido = new List<Pedido>();
            while (reader.Read())
            {
                //Cria um objeto Pedido
                Pedido pedido = new Pedido();
                pedido.PedidoID = Convert.ToInt16(reader["pedidoID"]);
                pedido.Data = Convert.ToDateTime(reader["data"]);
                pedido.Valor = Convert.ToDecimal(reader["valor"]);
                pedido.ClienteID = Convert.ToInt16(reader["clienteID"]);
                listaPedido.Add(pedido);
            }
            return listaPedido;
        }
        catch (Exception ex)
        {
            throw new Exception("Falha na operação: " + ex.Message);
        }
        finally
        {
            con.Close();
        }
    }
    public int GetUtimoPedido()
    {
        con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

        try
        {
            con.Open();
            cmd = new SqlCommand("UtimoPedido", con);
            cmd.CommandType = CommandType.StoredProcedure;

            //Executa o comando que retorna o valor do útimo pedido
            if (cmd.ExecuteScalar()==null)
                return 0;
            else
            {
                return Convert.ToInt16( cmd.ExecuteScalar());
            }
        }
        catch (Exception ex)
        {
            throw new Exception("Falha na operação: " + ex.Message);
        }
        finally
        {
            con.Close();
        }
    }
    public List<Pedido> GetPedidosComItens(int clienteID)
    {
        //Realiza uma consulta para pedidos usando a procedure GetPedidos.
        con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);
```

```
            SqlCommand cmd = new SqlCommand("GetPedidos", con);
            cmd.CommandType = CommandType.StoredProcedure;
            //cmd.Parameters.AddWithValue("@clienteID", clienteID);

            //Armazena o resulta em um DataSet temporário
            SqlDataAdapter adapter = new SqlDataAdapter(cmd);
            DataSet ds = new DataSet();
            adapter.Fill(ds, "Pedidos");

            //Realiza uma consulta para itensPedido usando o procedure Get.
            cmd.CommandText = "GetItensPedido";
            adapter.Fill(ds, "ItensPedido");

            //Configura uma relação entre as duas tabelas.
            //Isto torna mais fácil descobrir os itens em cada pedido.
            DataRelation relPedidoItens = new DataRelation("PedidoItens",
               ds.Tables["Pedidos"].Columns["PedidoID"],
               ds.Tables["ItensPedido"].Columns["PedidoID"]);
            ds.Relations.Add(relPedidoItens);

            //Constroi a coleção de objetos pedidos.
            List<Pedido> listaPedidos = new List<Pedido>();
            foreach (DataRow pedidoRow in ds.Tables["Pedidos"].Rows)
            {
               //Adicionar a coleção de aninhado objetos de itens para esta pedido.
               List<ItensPedido> ListaDeItens = new List<ItensPedido>();
               foreach (DataRow itensRow in pedidoRow.GetChildRows(relPedidoItens))
               {
                  ListaDeItens.Add(new ItensPedido(Convert.ToInt16(itensRow["itemNum"]),
                     Convert.ToInt16(itensRow["Qtdade"]),
                     Convert.ToDecimal(itensRow["precoVenda"]),
                     Convert.ToInt16(itensRow["pedidoID"]),
                     Convert.ToInt16(itensRow["produtoID"])));
               }
               listaPedidos.Add(new Pedido(Convert.ToInt16(pedidoRow["PedidoID"]),
                  ListaDeItens));
            }
            return listaPedidos;
         }
      }
   }
```

C# - implementação da classe ItensPedido

```
namespace SistemaPontoVenda
{
   public class ItensPedido
   {
      SqlConnection con = null;
      SqlCommand cmd = null;
```

```
private int itemNum;

public int ItemNum
{
   get { return itemNum; }
   set { itemNum = value; }
}
private int qtdade;

public int Qtdade
{
   get { return qtdade; }
   set { qtdade = value; }
}
private decimal precoVenda;

public decimal PrecoVenda
{
   get { return precoVenda; }
   set { precoVenda = value; }
}
private int pedidoID;

public int PedidoID
{
   get { return pedidoID; }
   set { pedidoID = value; }
}
private int produtoID;

public int ProdutoID
{
   get { return produtoID; }
   set { produtoID = value; }
}

public ItensPedido(int itemNum, int qtdade, decimal precovenda,
   int pedidoID, int produtoID)
{
   this.ItemNum = itemNum;
   this.Qtdade = qtdade;
   this.PrecoVenda = precovenda;
   this.PedidoID = pedidoID;
   this.ProdutoID = produtoID;
}

public ItensPedido() { }

public void InserirItensPedido(ItensPedido item)
{
   con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

   try
   {
      con.Open();
```

```
            cmd = new SqlCommand("InsertItensPedido", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.AddWithValue("@itemNum", item.ItemNum);
            cmd.Parameters.AddWithValue("@qtade", item.Qtdade);
            cmd.Parameters.AddWithValue("@precoVenda", item.PrecoVenda);
            cmd.Parameters.AddWithValue("@pedidoID", item.PedidoID);
            cmd.Parameters.AddWithValue("@produtoID", item.ProdutoID);

            //Executa o comando para inserir o Item pedido no banco de dado
            cmd.ExecuteNonQuery();
        }
        catch (Exception ex)
        {
            throw new Exception("Falha na operação: " + ex.Message);
        }
        finally
        {
            con.Close();
        }
    }
    public List<ItensPedido> GetItensPorPedidoID(int pedidoID)
    {
        con = new SqlConnection(ConfigurationManager.AppSettings["conString"]);

        try
        {
            con.Open();
            cmd = new SqlCommand("GetItensPedidoPorPedidoID", con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.AddWithValue("@pedidoID", pedidoID);

            //Executa o comando para recuperar os Itens o pedido no banco de dado
            SqlDataReader reader = cmd.ExecuteReader();
            //Cria uma lista para receber os registros dos itensPedido
            List<ItensPedido> listaItens = new List<ItensPedido>();
            while (reader.Read())
            {
                //Cria uma instância do objeto ItensPedido para receber o item retornado
                ItensPedido item = new ItensPedido();
                item.ItemNum = Convert.ToInt16(reader["itemNum"]);
                item.Qtdade = Convert.ToInt16(reader["qtdade"]);
                item.PrecoVenda = Convert.ToDecimal(reader["precoVenda"]);
                item.PedidoID = Convert.ToInt16(reader["pedidoID"]);
                item.ProdutoID = Convert.ToInt16(reader["produtoID"]);
                listaItens.Add(item);
            }
            return listaItens;
        }
        catch (Exception ex)
        {
            throw new Exception("Falha na operação: " + ex.Message);
        }
        finally
        {
            con.Close();
```

```
        }
    }
  }
}
```

Tela principal do sistema

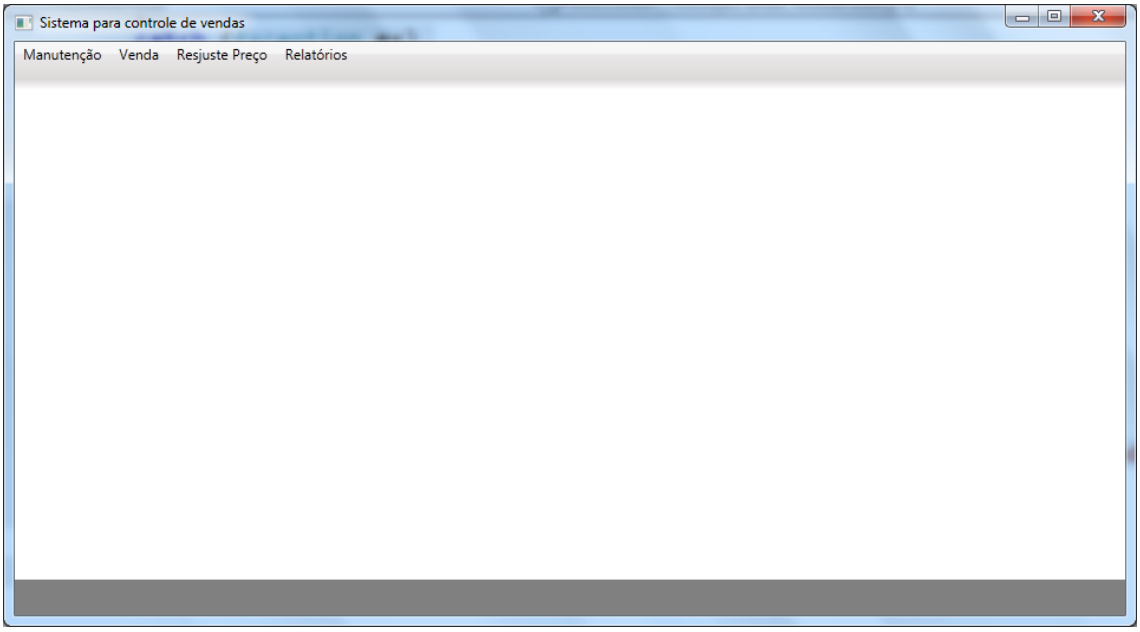

Figura 3: Tela principal do Sistema Ponto de vendas

XAML da Tela Principal do Sistema

```
x:Class="SistemaPontoVenda.MainWindow"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="Sistema para controle de vendas" Width="Auto" WindowState="Maximized"
WindowStyle="ThreeDBorderWindow">
  <Grid>
    <DockPanel Height="40" VerticalAlignment="Top" >
      <Menu DockPanel.Dock="Top" VerticalAlignment="Top" Height="40">
        <MenuItem Header="Manutenção">
          <MenuItem Header="_Cliente" Name="mCliente" Click="mCliente_Click"/>
          <MenuItem Header="_Produto" Name="mProduto" Click="mProduto_Click"/>
          <Separator />
          <MenuItem Header="_Sair" Name="mSair" Click="mSair_Click"/>
        </MenuItem>
        <MenuItem Header="Venda">
          <MenuItem Header="Realizar _Venda" Name="mRealiazarVenda"
Click="mRealiazarVenda_Click"></MenuItem>
        </MenuItem>
        <MenuItem Header="Resjuste Preço">
```

```
            <MenuItem Header="Reajustar preço do produto" Name="mReajprecoProduto"
Click="mReajprecoProduto_Click"></MenuItem>
          </MenuItem>
          <MenuItem Header="Relatórios">
            <MenuItem Header="Relação de clientes"  Name="mRelClientes"
Click="mRelClientes_Click"></MenuItem>
            <MenuItem Header="Catálogo de Produtos" Name="mCatalogoProduto"
Click="mCatalogoProduto_Click"></MenuItem>
            <MenuItem Header="Relatório de vendas"  Name="mRelVendas"
Click="mRelVendas_Click"></MenuItem>
          </MenuItem>
        </Menu>
      </DockPanel>
      <DockPanel Height="30" VerticalAlignment="Bottom" Background="Gray">
        <TextBlock Name="lblInfo" ></TextBlock>
      </DockPanel>
    </Grid>
</Window>
```

C# da tela principal do sistema

```
namespace SistemaPontoVenda
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }

        private void mCliente_Click(object sender, RoutedEventArgs e)
        {
            clsCliente cliente = new clsCliente();
            cliente.Owner = this;
            cliente.Show();
        }

        private void mSair_Click(object sender, RoutedEventArgs e)
        {
            this.Close();
        }

        private void mProduto_Click(object sender, RoutedEventArgs e)
        {
            jProduto jproduto = new jProduto();
            jproduto.Owner = this;
            jproduto.Show();
        }
```

```
        private void mRealiazarVenda_Click(object sender, RoutedEventArgs e)
        {
            jRealizarVenda jrealizarVenda = new jRealizarVenda();
            jrealizarVenda.Owner = this;
            jrealizarVenda.Show();
        }

        private void mReajprecoProduto_Click(object sender, RoutedEventArgs e)
        {
            jReajustarPreco jreajustarPreco = new jReajustarPreco();
            jreajustarPreco.Owner = this;
            jreajustarPreco.Show();
        }

        private void mRelClientes_Click(object sender, RoutedEventArgs e)
        {
            jRelacaoClientes jRelClientes = new jRelacaoClientes();
            jRelClientes.Owner = this;
            jRelClientes.Show();
        }

        private void mCatalogoProduto_Click(object sender, RoutedEventArgs e)
        {
            jCatalogoProduto jcatalogo = new jCatalogoProduto();
            jcatalogo.Owner = this;
            jcatalogo.Show();
        }

        private void mRelVendas_Click(object sender, RoutedEventArgs e)
        {
            jRelVendas jRelvendas = new jRelVendas();
            jRelvendas.Owner = this;
            jRelvendas.Show();
        }
    }
}
```

Janela Cadastrar Cliente

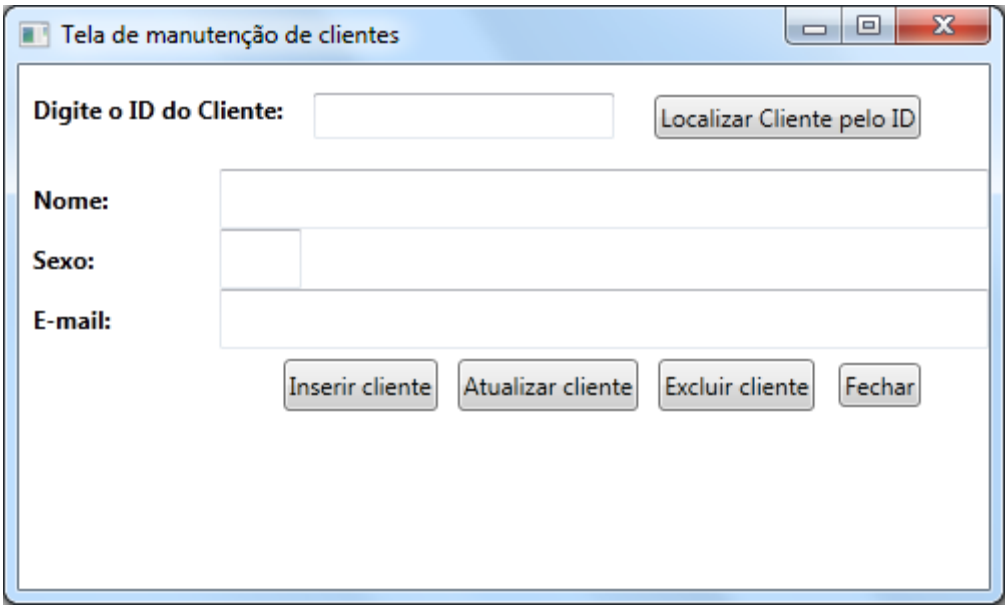

Figura 4: Janela para cadastro de clientes

XAML da janela Cadastrar Cliente

```
<Window x:Class="SistemaPontoVenda.clsCliente"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:dataInput="clr-
namespace:System.Windows.Input.Manipulations;assembly=System.Windows.Input.Manipulat
ions"
      Title="Tela de manutenção de clientes" Height="300" Width="500"
      WindowStartupLocation="CenterOwner" >
   <Grid>
      <Grid.ColumnDefinitions>
         <ColumnDefinition Width="100"/>
         <ColumnDefinition />
      </Grid.ColumnDefinitions>
      <Grid.RowDefinitions>
         <RowDefinition Height="Auto"/>
         <RowDefinition Height="Auto"/>
         <RowDefinition Height="Auto"/>
         <RowDefinition Height="Auto"/>
         <RowDefinition Height="Auto"/>
         <RowDefinition Height="Auto"/>
      </Grid.RowDefinitions>
      <StackPanel Orientation="Horizontal" Grid.ColumnSpan="2">
          <TextBlock Margin="7" Height="23" Text="Digite o ID do Cliente: "
FontWeight="Bold"></TextBlock>
          <TextBox Name="txtClienteID" Margin="5" Height="23" Width="150"
                Text="{Binding ClienteID, Mode=TwoWay, ValidatesOnExceptions=True,
NotifyOnValidationError=True}"
                Visibility="Visible"></TextBox>
         <Button Margin="15" Name="GetCliente" Content="Localizar Cliente pelo ID"
Click="GetCliente_Click"></Button>
```

```
        </StackPanel>
        <TextBlock Margin="7" Grid.Row="1" FontWeight="Bold">Nome:</TextBlock>
        <TextBox Name="txtNome" Grid.Row="1" Grid.Column="1"
                 Text="{Binding Nome, ValidatesOnExceptions=True,
NotifyOnValidationError=True}"/>
        <TextBlock Margin="7" Grid.Row="2" FontWeight="Bold">Sexo:</TextBlock>
        <TextBox Name="txtSexo" Grid.Row="2" Grid.Column="1"
                 Text="{Binding Sexo, ValidatesOnExceptions=True,
NotifyOnValidationError=True}"
                 Margin="0,0,343,0" />
        <TextBlock Margin="7" Grid.Row="3" FontWeight="Bold">E-mail:</TextBlock>
        <TextBox Name="txtEmail" Grid.Row="3" Grid.Column="1"
                 Text="{Binding Email, ValidatesOnExceptions=True,
NotifyOnValidationError=True, Mode=TwoWay}"/>
        <StackPanel Orientation="Horizontal" Grid.Row="4" Grid.Column="1"
HorizontalAlignment="Center">
            <Button Margin="5" Name="btnInserirCliente" Content="Inserir cliente"
Click="btnInserirCliente_Click"/>
            <Button Margin="5" Name="btnAtualizarCliente" Content="Atualizar cliente"
Click="btnAtualizarCliente_Click" />
            <Button Margin="5" Name="btnExcluirCliente" Content="Excluir cliente"
Click="btnExcluirCliente_Click" />
            <Button Margin="7" Name="btnFechar" Content="Fechar" Click="btnFechar_Click"/>
        </StackPanel>

    </Grid>
</Window>
```

C# da janela cadastrar cliente

```
namespace SistemaPontoVenda
{
    /// <summary>
    /// Interaction logic for Cliente.xaml
    /// </summary>
    public partial class clsCliente : Window
    {
        public clsCliente()
        {
            InitializeComponent();
        }

        private void btnInserirCliente_Click(object sender, RoutedEventArgs e)
        {
            clsCliente cli = new clsCliente();

            cli.ClienteID = Convert.ToInt16(txtClienteID.Text);
            cli.Nome = txtNome.Text;
            cli.Sexo = txtSexo.Text;
            cli.Email = txtEmail.Text;
```

```
            //Chama o método para inserir um cliente no banco de dados.
            cli.InserirCliente(cli);
            MessageBox.Show("Registro inserido com sucesso.");
        }

        private void btnFechar_Click(object sender, RoutedEventArgs e)
        {
            this.Close();
        }

        private void GetCliente_Click(object sender, RoutedEventArgs e)
        {
            clsCliente cli = new clsCliente(); //Este cliente é para chamar o método
LocalizarClientePorID
            clsCliente clienteRerornado = new clsCliente(); //Este cliente recebe o cliente retornado

            cli.ClienteID = Convert.ToInt16(txtClienteID.Text);

            clienteRerornado = cli.RetornarUmCliente(cli);

            txtNome.Text = clienteRerornado.Nome;
            txtSexo.Text = clienteRerornado.Sexo;
            txtEmail.Text = clienteRerornado.Email;
        }

        private void btnAtualizarCliente_Click(object sender, RoutedEventArgs e)
        {
            clsCliente cli = new clsCliente();

            cli.ClienteID = Convert.ToInt16(txtClienteID.Text);
            cli.Nome = txtNome.Text;
            cli.Sexo = txtSexo.Text;
            cli.Email = txtEmail.Text;

            //Chama o método para alterar um cliente no banco de dados.
            cli.AlterarCliente(cli);
            MessageBox.Show("Registro alterado com sucesso.");
        }

        private void btnExcluirCliente_Click(object sender, RoutedEventArgs e)
        {
            clsCliente cli = new clsCliente();

            cli.ClienteID = Convert.ToInt16(txtClienteID.Text);

            //Chama o método para excluir um cliente no banco de dados.
            cli.ExcluirCliente(cli);
            MessageBox.Show("Registro excluído com sucesso.");
        }
    }
}
```

Janela para cadastro de produtos

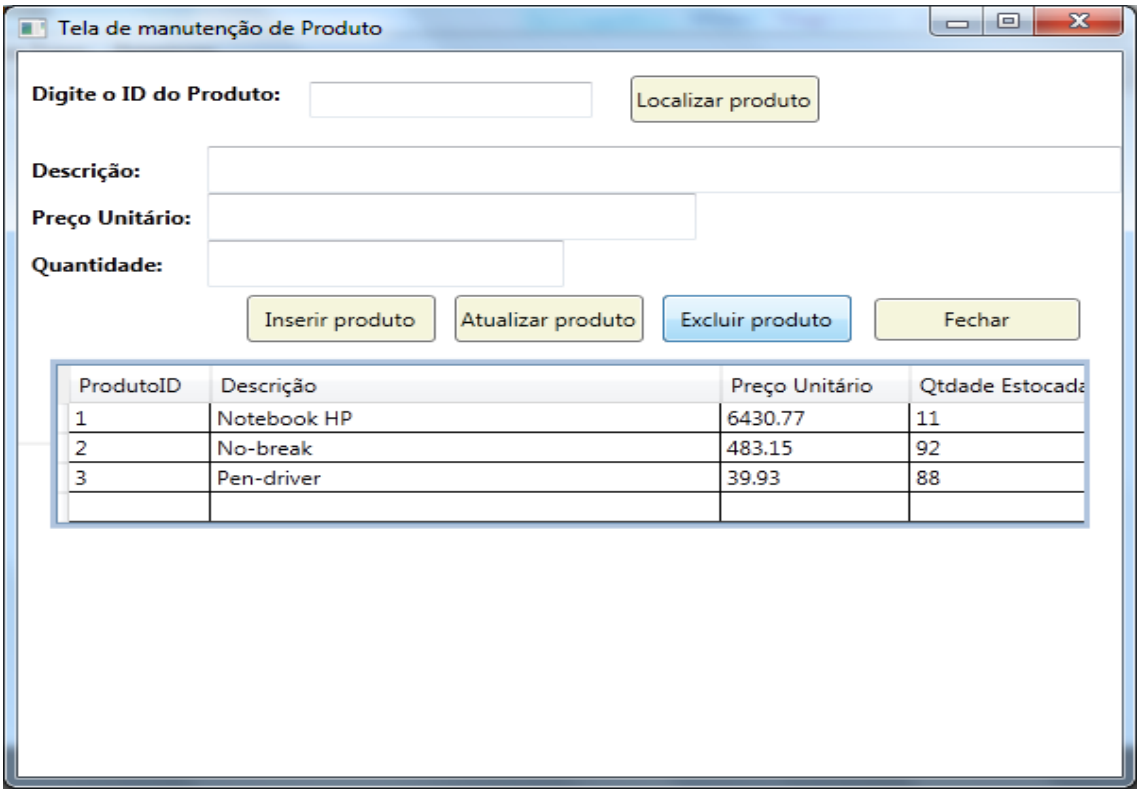

Figura 5: Janela para cadastro de produtos

XAML da janela Cadastrar Produto

```
<Window x:Class="SistemaPontoVenda.jProduto"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="Tela de manutenção de Produto" Height="500" Width="600"
      WindowStartupLocation="CenterOwner" Loaded="Window_Loaded">
   <Window.Resources>
      <Style x:Key="estiloButtom" TargetType="Button">
         <Setter Property="Background" Value="Beige"></Setter>
         <Setter Property="FontSize" Value="12"></Setter>
         <Setter Property="Height" Value="30"></Setter>
         <Setter Property="Width" Value="100"></Setter>
      </Style>
   </Window.Resources>
   <Grid>
      <Grid.ColumnDefinitions>
         <ColumnDefinition Width="100"/>
         <ColumnDefinition />
      </Grid.ColumnDefinitions>
      <Grid.RowDefinitions>
         <RowDefinition Height="Auto"/>
         <RowDefinition Height="Auto"/>
         <RowDefinition Height="Auto"/>
         <RowDefinition Height="Auto"/>
         <RowDefinition Height="Auto"/>
         <RowDefinition Height="Auto"/>
```

```
    </Grid.RowDefinitions>
    <StackPanel Orientation="Horizontal" Grid.ColumnSpan="2">
      <TextBlock Margin="7" Height="23" Text="Digite o ID do Produto: "
FontWeight="Bold"></TextBlock>
      <TextBox Name="txtProdutoID" Margin="5" Height="23" Width="150"></TextBox>
      <Button Margin="15" Name="GetProduto" Content="Localizar produto pelo ID"
          Style="{StaticResource estiloButtom}" Click="GetProduto_Click"></Button>
    </StackPanel>
    <TextBlock Margin="7" Grid.Row="1" FontWeight="Bold">Descrição:</TextBlock>
    <TextBox Name="txtDescricao" Grid.Row="1" Grid.Column="1" Text="{Binding
Descricao}"/>
    <TextBlock Margin="7" Grid.Row="2" FontWeight="Bold">Preço Unitário:</TextBlock>
    <TextBox Name="txtPreco" Grid.Row="2" Grid.Column="1" Text="{Binding
PrecoUnitario}" Margin="0,0,225,0" />
    <TextBlock Margin="7" Grid.Row="3" FontWeight="Bold">Quantidade:</TextBlock>
    <TextBox Name="txtEstocada" Grid.Row="3" Grid.Column="1" Text="{Binding
Estocada}" Margin="0,0,295,0" />
    <StackPanel Orientation="Horizontal" Grid.Row="4" Grid.Column="1"
HorizontalAlignment="Center">
      <Button Margin="5" Name="btnInserirProduto" Style="{StaticResource estiloButtom}"
          Content="Inserir produto" Click="btnInserirProduto_Click"/>
      <Button Margin="5" Name="btnAtualizarProduto" Style="{StaticResource
estiloButtom}"
          Content="Atualizar produto" Click="btnAtualizarProduto_Click"/>
      <Button Margin="5" Name="btnExcluirProduto" Content="Excluir produto"
Style="{StaticResource estiloButtom}"
          Click="btnExcluirProduto_Click"/>
      <Button Margin="7" Name="btnFechar" Content="Fechar" Style="{StaticResource
estiloButtom}"
          Click="btnFechar_Click" Height="27" Width="109" />
    </StackPanel>
    <GridSplitter Grid.Row="5" HorizontalAlignment="Stretch" VerticalAlignment="Center"
Height="2"/>
    <Border Grid.Row="5" Grid.ColumnSpan="2" Padding="3" Margin="5"
Background="LightSteelBlue" Width="550">
      <Grid x:Name="gridProducts">
        <Grid.ColumnDefinitions>
          <ColumnDefinition Width="Auto"></ColumnDefinition>
          <ColumnDefinition></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
          <RowDefinition Height="Auto"></RowDefinition>
        </Grid.RowDefinitions>
        <DataGrid Name="dataGridProduto" AutoGenerateColumns="False" >
          <DataGrid.Columns>
            <DataGridTextColumn Header="ProdutoID" Width="75" Binding="{Binding
ProdutoID}"></DataGridTextColumn>
            <DataGridTextColumn Header="Descrição" Width="270" Binding="{Binding
Descricao}"></DataGridTextColumn>
            <DataGridTextColumn Header="Preço Unitário" Width="100"
Binding="{Binding PrecoUnitario}"></DataGridTextColumn>
            <DataGridTextColumn Header="Qtdade Estocada" Width="100"
Binding="{Binding Estocada}"></DataGridTextColumn>
          </DataGrid.Columns>
        </DataGrid>
```

```
        </Grid>
      </Border>
   </Grid>
</Window>
```

C# da janela Cadastrar produto

```
namespace SistemaPontoVenda
{
   /// <summary>
   /// Interaction logic for jProduto.xaml
   /// </summary>
   public partial class jProduto : Window
   {
      public jProduto()
      {
         InitializeComponent();
      }

      private void btnFechar_Click(object sender, RoutedEventArgs e)
      {
         this.Close();
      }

      private void btnInserirProduto_Click(object sender, RoutedEventArgs e)
      {
         Produto produto = new Produto();
         produto.ProdutoID = Convert.ToInt16(txtProdutoID.Text);
         produto.Descricao = txtDescricao.Text;
         produto.PrecoUnitario = Convert.ToDecimal(txtPreco.Text);
         produto.Estocada = Convert.ToInt16(txtEstocada.Text);

         try
         {
            produto.InserirProduto(produto);
            MessageBox.Show("Produto inserido com sucesso.");
            dataGridProduto.ItemsSource = produto.RetornarTodosProdutos();
         }
         catch (Exception ex)
         {
            MessageBox.Show("Falha na operação." + ex.Message);
         }
      }

      private void Window_Loaded(object sender, RoutedEventArgs e)
      {
         Produto produto = new Produto();
         dataGridProduto.ItemsSource = produto.RetornarTodosProdutos();
      }

      private void GetProduto_Click(object sender, RoutedEventArgs e)
```

```
        {
            Produto produto = new Produto();
            produto = (Produto)dataGridProduto.SelectedItem;

            Produto rproduto = new Produto();
            rproduto = produto.LocalizarProdutoPorID(produto);

            //Exibir os dados nas text boxs
            txtProdutoID.Text = rproduto.ProdutoID.ToString();
            txtDescricao.Text = rproduto.Descricao;
            txtPreco.Text = rproduto.PrecoUnitario.ToString();
            txtEstocada.Text = rproduto.Estocada.ToString();
        }

        private void btnAtualizarProduto_Click(object sender, RoutedEventArgs e)
        {
            Produto produto = new Produto();
            produto.ProdutoID = Convert.ToInt16(txtProdutoID.Text);
            produto.Descricao = txtDescricao.Text;
            produto.PrecoUnitario = Convert.ToDecimal(txtPreco.Text);
            produto.Estocada = Convert.ToInt16(txtEstocada.Text);

            try
            {
                produto.AlterarProduto(produto);
                MessageBox.Show("Produto alterado com sucesso.");
                dataGridProduto.ItemsSource = produto.RetornarTodosProdutos();
            }
            catch (Exception ex)
            {
                MessageBox.Show("Falha na operação." + ex.Message);
            }
        }

        private void btnExcluirProduto_Click(object sender, RoutedEventArgs e)
        {
            Produto produto = new Produto();
            produto = (Produto)dataGridProduto.SelectedItem;

            try
            {
                produto.ExcluirProduto(produto);
                MessageBox.Show("Produto excluído com sucesso.");
                dataGridProduto.ItemsSource = produto.RetornarTodosProdutos();
            }
            catch (Exception ex)
            {
                MessageBox.Show("Falha na operação." + ex.Message);
            }
        }
    }
}
```

Janela para reajustar preços dos produtos

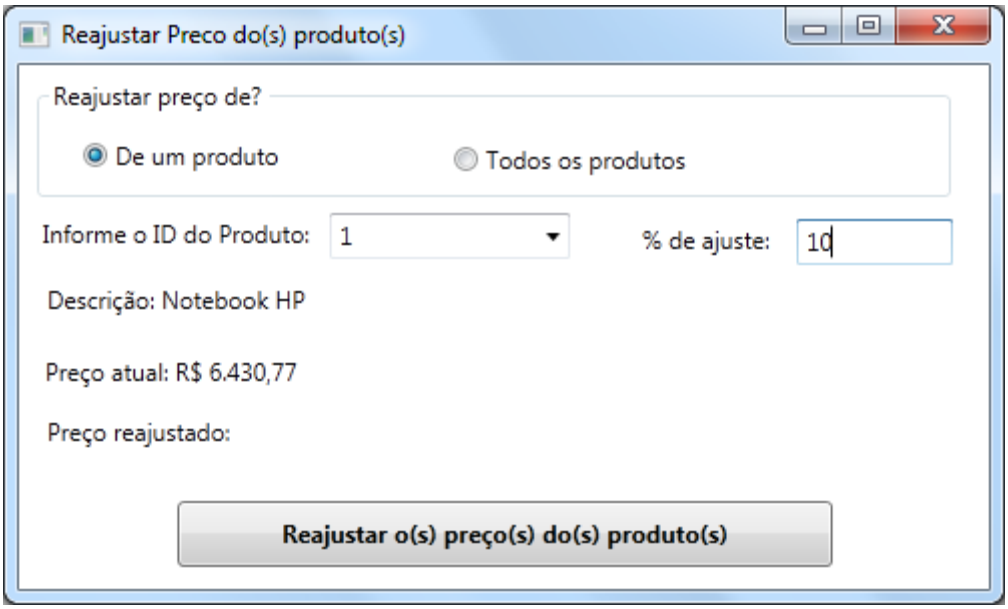

Figura 6: Janela para reajustar preços dos produtos

XAML da janela reajustar preço

```
<Window x:Class="SistemaPontoVenda.jReajustarPreco"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Reajustar Preco do(s) produto(s)" Height="300" Width="500"
        WindowStartupLocation="CenterOwner">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="67*" />
            <RowDefinition Height="68*" />
            <RowDefinition Height="66*" />
            <RowDefinition Height="60*" />
        </Grid.RowDefinitions>
        <GroupBox Header="Reajustar preço de?" Height="61" HorizontalAlignment="Left"
Margin="8,6,0,0" Name="ReajPreco" VerticalAlignment="Top" Width="458">
            <Grid>
                <RadioButton Content="De um produto" Height="16" HorizontalAlignment="Left"
Margin="18,14,0,0" Name="radioButton1" VerticalAlignment="Top"
GroupName="ReajPreco" Checked="radioButton1_Checked" />
                <RadioButton Content="Todos os produtos" Height="16"
HorizontalAlignment="Left" Margin="203,16,0,0" Name="radioButton2"
VerticalAlignment="Top" IsChecked="False" GroupName="ReajPreco"
Checked="radioButton2_Checked" IsHitTestVisible="True" />
            </Grid>
        </GroupBox>
        <TextBlock Grid.Row="1" Height="23" HorizontalAlignment="Left" Margin="12,9,0,0"
Name="textBlock1" Text="Informe o ID do Produto:" VerticalAlignment="Top" />
        <ComboBox Grid.Row="1" Height="23" HorizontalAlignment="Left"
Margin="155,7,0,0" Name="cboProduto" VerticalAlignment="Top" Width="120"
IsEditable="True" IsEnabled="False" DisplayMemberPath="ProdutoID"
SelectionChanged="cboProduto_SelectionChanged" />
```

```
        <TextBlock Grid.Row="1" Height="23" HorizontalAlignment="Left"
Margin="311,12,0,0" Name="textBlock2" Text="% de ajuste:" VerticalAlignment="Top" />
        <TextBox Grid.Row="1" Height="23" HorizontalAlignment="Left" Margin="388,10,0,0"
Name="txtPercentual" VerticalAlignment="Top" Width="78" />
        <TextBlock Grid.Row="2" Height="23" HorizontalAlignment="Left" Margin="13,10,0,0"
Name="tbxPrecoAtual" Text="Preço atual:" VerticalAlignment="Top" Width="447" />
        <TextBlock Grid.Row="2" Height="23" HorizontalAlignment="Left" Margin="14,40,0,0"
Name="tbxPrecoReajustado" Text="Preço reajustado: " VerticalAlignment="Top"
Width="446" />
        <TextBlock Grid.Row="1" Height="23" HorizontalAlignment="Left" Margin="14,42,0,0"
Name="tbxDescricaoProduto" Text="Descrição do produto:" VerticalAlignment="Top" />
        <Button Content="Reajustar o(s) preço(s) do(s) produto(s)" Grid.Row="3" Height="33"
HorizontalAlignment="Left" Margin="79,16,0,0" Name="btnReajustarPreco"
VerticalAlignment="Top" Width="327" FontWeight="Bold" Click="btnReajustarPreco_Click"
/>
    </Grid>
</Window>
```

C# da janela reajustar preço

```
namespace SistemaPontoVenda
{
    /// <summary>
    /// Interaction logic for jReajustarPreco.xaml
    /// </summary>
    public partial class jReajustarPreco : Window
    {
        ComboBox combo = new ComboBox();

        public jReajustarPreco()
        {
            InitializeComponent();
        }

        private void radioButton1_Checked(object sender, RoutedEventArgs e)
        {
            Produto produto = new Produto();
            cboProduto.ItemsSource = produto.RetornarTodosProdutos();
            if (radioButton1.IsChecked == true)
                cboProduto.IsEnabled = true;
            else
                cboProduto.IsEnabled = false;
        }

        private void cboProduto_SelectionChanged(object sender, SelectionChangedEventArgs e)
        {
            Produto produto = new Produto();
            produto = (Produto)cboProduto.SelectedItem;
            tbxDescricaoProduto.Text = "Descrição: " + produto.Descricao;
            CultureInfo culture = new CultureInfo("pt-BR");
            decimal preco = produto.PrecoUnitario;
```

```
            tbxPrecoAtual.Text = "Preço atual: " + preco.ToString("C", culture);
        }

        private void radioButton2_Checked(object sender, RoutedEventArgs e)
        {
            cboProduto.IsEnabled = false;
        }

        private void btnReajustarPreco_Click(object sender, RoutedEventArgs e)
        {
            Produto produto = new Produto();

            try
            {
                if (radioButton1.IsChecked == true)
                {
                    produto = (Produto)cboProduto.SelectedItem;

                    produto.ReajustarPrecoDeUmProduto(produto.ProdutoID,
Convert.ToDecimal(txtPercentual.Text));
                    tbxPrecoReajustado.Text = (produto.PrecoUnitario * (1 +
Convert.ToDecimal(txtPercentual.Text) / 100)).ToString("C");
                }
                else if (radioButton2.IsChecked == true)
                {
                    produto.ReajustarPrecoDosProdutos(Convert.ToDecimal(txtPercentual.Text));
                    MessageBox.Show("Todos os preços foram reajustados com sucesso.");
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show("Falha na operação." + ex.Message);
            }
        }
    }
}
```

Janela para realizar vendas

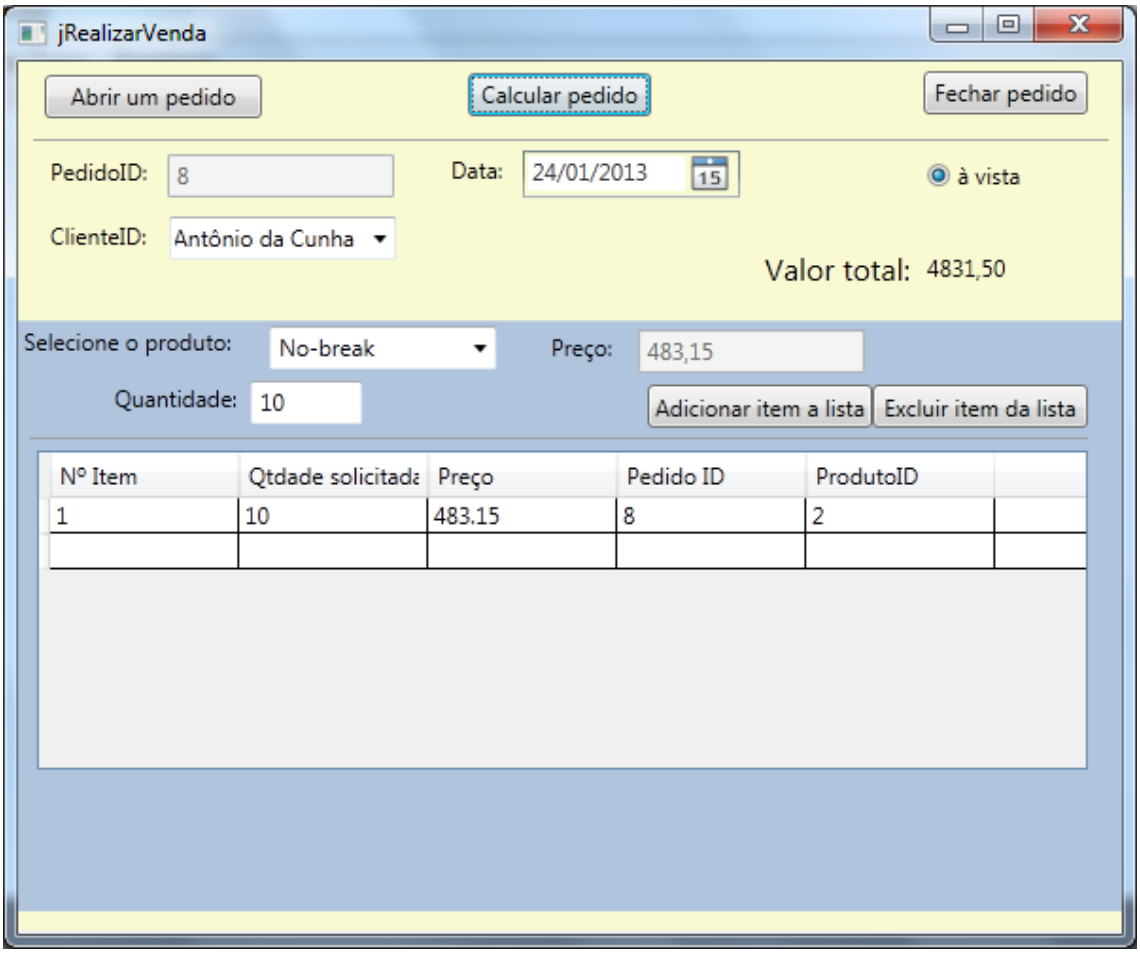

Figura 7: Janela vendas de produto

XAML da janela realizar vendas

```
<Window x:Class="SistemaPontoVenda.jRealizarVenda"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="jRealizarVenda" Height="500" Width="600"
      WindowStartupLocation="CenterOwner" Loaded="Window_Loaded">
   <Grid Background="LightGoldenrodYellow">
      <Grid.RowDefinitions>
         <RowDefinition Height="36*" />
         <RowDefinition Height="88*" />
         <RowDefinition Height="335*" />
         <RowDefinition Height="Auto"/>
      </Grid.RowDefinitions>

      <TextBlock Grid.Row="1" Height="23" HorizontalAlignment="Left" Margin="17,14,0,0"
Name="textBlock3" Text="PedidoID:" VerticalAlignment="Top" />
      <TextBox Grid.Row="1" Height="23" HorizontalAlignment="Left" Margin="79,13,0,0"
Name="txtPedido" VerticalAlignment="Top" Width="120" IsEnabled="False" />
      <TextBlock Grid.Row="1" Height="23" HorizontalAlignment="Left"
Margin="229,13,0,0" Name="textBlock4" Text="Data:" VerticalAlignment="Top" />
```

```
        <TextBlock Grid.Row="1" Height="23" HorizontalAlignment="Left" Margin="17,48,0,0"
Name="textBlock5" Text="ClienteID:" VerticalAlignment="Top" />

        <RadioButton Content="à vista" Grid.Row="1" Height="16" HorizontalAlignment="Left"
Margin="481,16,0,0" Name="radioButton1" VerticalAlignment="Top" IsChecked="True" />
        <TextBlock Grid.Row="1" Height="23" HorizontalAlignment="Left"
Margin="395,63,0,0" Name="textBlock6" Text="Valor total:" VerticalAlignment="Top"
FontSize="16" />
        <TextBlock Grid.Row="1" Height="23" HorizontalAlignment="Left"
Margin="481,65,0,0" Name="tbValorTotal" Text="TextBlock" VerticalAlignment="Top"
Width="85" />

        <GridSplitter Grid.Row="3" HorizontalAlignment="Stretch" VerticalAlignment="Center"
Height="2"/>

        <Border Grid.Row="2" Padding="3" Background="LightSteelBlue" Margin="0,13,0,10">
            <TextBlock Height="23" Text="Selecione o produto:" VerticalAlignment="Top" />
        </Border>
        <ComboBox Grid.Row="2" Height="23" HorizontalAlignment="Right"
Margin="0,16,331,0" Name="cboProduto" VerticalAlignment="Top" Width="120"
DisplayMemberPath="Descricao" IsEditable="True"
SelectionChanged="cboProduto_SelectionChanged" />
        <TextBlock Grid.Row="2" Height="23" HorizontalAlignment="Left" Margin="51,46,0,0"
Text="Quantidade:" VerticalAlignment="Top" />
        <TextBox Grid.Row="2" Height="23" HorizontalAlignment="Left" Margin="123,45,0,0"
Name="txtQtdade" VerticalAlignment="Top" Width="59" />
        <Button Content="Adicionar item a lista" Grid.Row="2" Height="23"
HorizontalAlignment="Right" Margin="0,47,132,0" Name="btnAdicionarItem"
VerticalAlignment="Top" Width="119" Click="btnAdicionarItem_Click" />
        <Separator Grid.Row="2" Height="10" HorizontalAlignment="Left" Margin="6,70,0,0"
Name="separator1" VerticalAlignment="Top" Width="560" Foreground="#FF36DE36" />
        <DataGrid AutoGenerateColumns="False" Grid.Row="2" Height="169"
HorizontalAlignment="Left" Margin="10,82,0,0" Name="dataItens" VerticalAlignment="Top"
Width="556">
            <DataGrid.Columns>
                <DataGridTextColumn Header="Nº Item" Binding="{Binding ItemNum}"
Width="100"></DataGridTextColumn>
                <DataGridTextColumn Header="Qtdade solicitada" Binding="{Binding Qtdade}"
Width="100"></DataGridTextColumn>
                <DataGridTextColumn Header="Preço" Binding="{Binding PrecoVenda}"
Width="100"></DataGridTextColumn>
                <DataGridTextColumn Header="Pedido ID" Binding="{Binding PedidoID}"
Width="100"></DataGridTextColumn>
                <DataGridTextColumn Header="ProdutoID" Binding="{Binding ProdutoID}"
Width="100"></DataGridTextColumn>
            </DataGrid.Columns>
        </DataGrid>
        <Button Content="Fechar pedido" Height="23" HorizontalAlignment="Left"
Margin="479,5,0,0" Name="btnFecharPedido" VerticalAlignment="Top" Width="87"
Click="btnFecharPedido_Click" />
        <Button Content="Calcular pedido" Height="23" HorizontalAlignment="Left"
Margin="238,6,0,0" Name="btnCalcularPedido" VerticalAlignment="Top" Width="97"
Click="btnCalcularPedido_Click" />
```

```
        <Separator Grid.Row="1" Height="4" HorizontalAlignment="Left" Margin="8,3,0,0"
Name="separator2" VerticalAlignment="Top" Width="569" />
        <Button Content="Abrir um pedido" Height="23" HorizontalAlignment="Left"
Margin="14,7,0,0" Name="btnAbrirPedido" VerticalAlignment="Top" Width="115"
Click="btnAbrirPedido_Click" />
        <DatePicker Grid.Row="1" Height="25" HorizontalAlignment="Left"
Margin="267,11,0,0" Name="datePicker1" VerticalAlignment="Top" Width="115" />
        <TextBlock Grid.Row="2" Height="23" HorizontalAlignment="Left"
Margin="282,19,0,0" Name="textBlock1" Text="Preço:" VerticalAlignment="Top" />
        <TextBox Grid.Row="2" Height="23" HorizontalAlignment="Left" Margin="328,18,0,0"
Name="txtPrecoVenda" VerticalAlignment="Top" Width="120" Text="{Binding
PrecoUnitario}" IsEnabled="False" />
        <ComboBox Grid.Row="1" Height="23" HorizontalAlignment="Left"
Margin="80,46,0,0" Name="cboClientes" VerticalAlignment="Top" Width="120"
DisplayMemberPath="Nome" IsEditable="True" IsReadOnly="True" />
        <Button Content="Excluir item da lista" Grid.Row="2" Height="23"
HorizontalAlignment="Left" Margin="452,47,0,0" Name="btnExcluirItem"
VerticalAlignment="Top" Width="114" Click="btnExcluirItem_Click" />
    </Grid>
</Window>
```

C# da janela realizar vendas

```
namespace SistemaPontoVenda
{
    /// <summary>
    /// Interaction logic for jRealizarVenda.xaml
    /// </summary>
    public partial class jRealizarVenda : Window
    {
        //Declarar uma lista de Itens.
        ObservableCollection<ItensPedido> listaItens = new
ObservableCollection<ItensPedido>();

        //Um contador para o número do pedido
        private int contador = 1;

        public jRealizarVenda()
        {
            InitializeComponent();
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            //Preecne a comobox como com os ID dos clientes
            clsCliente cliente = new clsCliente();
            cboClientes.ItemsSource = cliente.GetClientes();

            //Preence a comobox com a descrição dos produtos
            Produto produto = new Produto();
            cboProduto.ItemsSource = produto.RetornarTodosProdutos();
```

```
    }

    private void btnAbrirPedido_Click(object sender, RoutedEventArgs e)
    {
        //Usa o método GetUtimoPedido da Classe Pedido para ir ao banco e trazer o útimo
pedido.
        Pedido pedido = new Pedido();
        txtPedido.Text = (pedido.GetUtimoPedido() + 1).ToString();
    }

    private void btnAdicionarItem_Click(object sender, RoutedEventArgs e)
    {
        Produto produto = new Produto();

        //Pegar o ID do produto selecionado na cboProduto
        produto = (Produto)cboProduto.SelectedItem;

        ItensPedido item = new ItensPedido();
        item.ItemNum = contador;
        if (txtQtdade.Text == string.Empty)
        {
            MessageBox.Show("Por favor!, digite a quantidade.");
            txtQtdade.Focus();
        }
        else
        {
            item.Qtdade = Convert.ToInt16(txtQtdade.Text);
        }
        item.PrecoVenda = Convert.ToDecimal( txtPrecoVenda.Text);
        if (txtPedido.Text == string.Empty)
        {
            MessageBox.Show("Por favor!, informe o número do pedido.");
            txtPedido.Focus();
        }
        else
        {
            item.PedidoID = Convert.ToInt16( txtPedido.Text);
        }
        item.ProdutoID = produto.ProdutoID;

        listaItens.Add(item);
        dataItens.ItemsSource = listaItens;

        //Incrementa o contador
        contador += 1;
    }

    private void cboProduto_SelectionChanged(object sender, SelectionChangedEventArgs e)
    {
        Produto produto = new Produto();

        produto = (Produto)cboProduto.SelectedItem;
        txtPrecoVenda.Text = produto.PrecoUnitario.ToString();
    }
```

```
        private void btnCalcularPedido_Click(object sender, RoutedEventArgs e)
        {
            Decimal valorTotalPedido = 0;
            foreach (ItensPedido item in listaItens)
            {
                valorTotalPedido += item.PrecoVenda * item.Qtdade;
            }
            tbValorTotal.Text = valorTotalPedido.ToString();
        }

        private void btnFecharPedido_Click(object sender, RoutedEventArgs e)
        {
            Pedido pedido = new Pedido();
            pedido.PedidoID = Convert.ToInt16(txtPedido.Text);
            pedido.Data = Convert.ToDateTime(datePicker1.SelectedDate);
            pedido.Valor = Convert.ToDecimal(tbValorTotal.Text);

            //Instancia um cliente para poder recuperar o seu ID da combobox, onde está o cliente
selecionado
            clsCliente cliente = new clsCliente();
            cliente = (clsCliente)cboClientes.SelectedItem;

            pedido.ClienteID = cliente.ClienteID;

            try
            {
                //Inserir o pedido no banco de dados.
                pedido.InserirPedido(pedido);

                //Instancia um itempedido para que possa ser inserido.
                ItensPedido newItem = new ItensPedido();
                //Percorre toda a lista de itens da DataGrid.
                foreach (ItensPedido item in listaItens)
                {
                    //Atualizar as propriedades do novo item com as propriedades do item lido da lista.
                    newItem.ItemNum = item.ItemNum;
                    newItem.Qtdade = item.Qtdade;
                    newItem.PrecoVenda = item.PrecoVenda;
                    newItem.PedidoID = item.PedidoID;
                    newItem.ProdutoID = item.ProdutoID;
                    //Chama o método inserirItem para gravar os dados do item no banco de dados.
                    newItem.InserirItensPedido(newItem);

                    //Chama o método DarBaixaNoEstoque para atualizar a quantidade estocada na
tabela Produto
                    Produto produto = new Produto();
                    produto.DarBaixaNoProduto(newItem.ProdutoID, newItem.Qtdade);
                }
                MessageBox.Show("Operação realizada com sucesso.");
                txtPedido.Focus();
            }
            catch (Exception ex)
            {
                MessageBox.Show("Erro na operação." + ex.Message );
            }
```

```
        }

        private void btnExcluirItem_Click(object sender, RoutedEventArgs e)
        {
            if (dataItens.SelectedItem != null)
            {
                listaItens.Remove((ItensPedido)dataItens.SelectedItem);
            }
            else
            {
                MessageBox.Show("Atenção!, você precisa selecionar um item para poder excluir.");
            }
        }

    }
}
```

Relatório de relação de clientes cadastrados

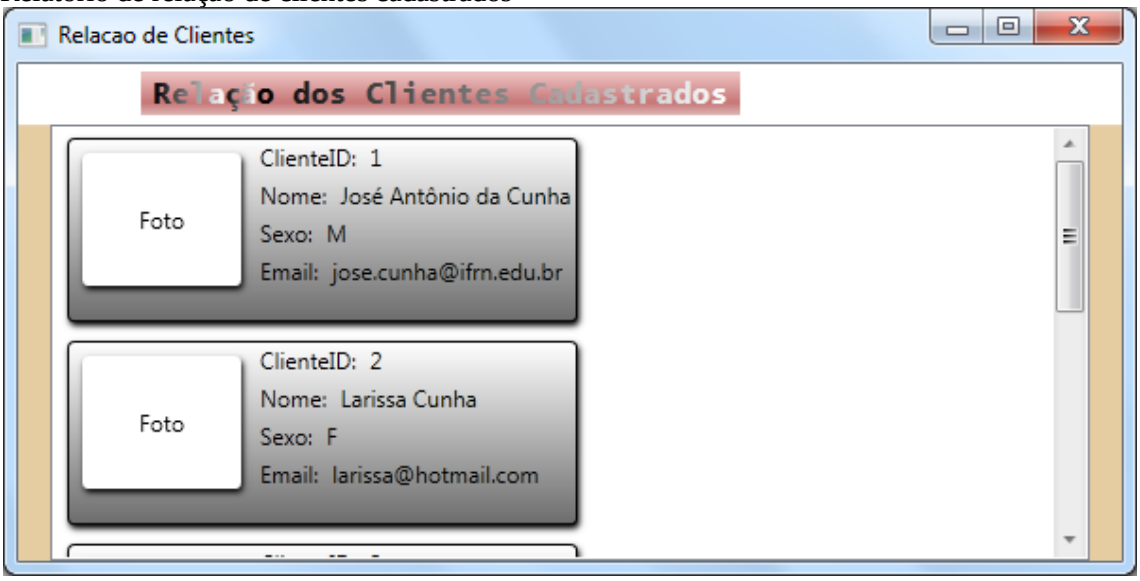

Figura 8: Relação dos clientes cadastrados

XAML da janela relação de Cliente

```
<Window x:Class="SistemaPontoVenda.jRelacaoClientes"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Relacao de Clientes" Height="300" Width="600"
WindowStartupLocation="CenterOwner"
        Loaded="Window_Loaded">
    <Window.Resources>
        <ItemsPanelTemplate x:Key="ItemTemplate">
            <WrapPanel />
        </ItemsPanelTemplate>
        <DataTemplate x:Key="listBoxTemplate">
            <Grid Width="270" Margin="5">
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width="100" />
```

```
                <ColumnDefinition />
            </Grid.ColumnDefinitions>
            <Grid.RowDefinitions>
                <RowDefinition Height="Auto"></RowDefinition>
                <RowDefinition Height="Auto"></RowDefinition>
                <RowDefinition Height="Auto"></RowDefinition>
                <RowDefinition Height="Auto"></RowDefinition>
                <RowDefinition Height="Auto"></RowDefinition>
            </Grid.RowDefinitions>
            <Rectangle Stroke="Black" RadiusX="3" RadiusY="3" Grid.RowSpan="5"
Grid.ColumnSpan="2">
                <Rectangle.Fill>
                    <LinearGradientBrush StartPoint="0,0" EndPoint="0,1">
                        <GradientStop Color="White" Offset="0"/>
                        <GradientStop Color="#6D6D6D" Offset="1"/>
                    </LinearGradientBrush>
                </Rectangle.Fill>
                <Rectangle.Effect>
                    <DropShadowEffect ShadowDepth="2"/>
                </Rectangle.Effect>
            </Rectangle>
            <Border BorderBrush="Black" Background="White" Margin="8" Height="70"
                    VerticalAlignment="Top" CornerRadius="3" Grid.RowSpan="5">
                <Border.Effect>
                    <DropShadowEffect ShadowDepth="2"/>
                </Border.Effect>
            <TextBlock Text="Foto" VerticalAlignment="Center"
HorizontalAlignment="Center" />
        </Border>
        <StackPanel Orientation="Horizontal" Grid.Row="0" Grid.Column="1">
            <TextBlock Text="ClienteID: " Margin="2"/>
            <TextBlock Text="{Binding ClienteID}" Margin="2"></TextBlock>
        </StackPanel>
        <StackPanel Orientation="Horizontal" Grid.Row="1" Grid.Column="1">
            <TextBlock Text="Nome: " Margin="2"></TextBlock>
            <TextBlock Text="{Binding Nome}" Margin="2"></TextBlock>
        </StackPanel>
        <StackPanel Orientation="Horizontal" Grid.Row="2" Grid.Column="1">
            <TextBlock Text="Sexo: " Margin="2"></TextBlock>
            <TextBlock Text="{Binding Sexo}" Margin="2"></TextBlock>
        </StackPanel>
        <StackPanel Orientation="Horizontal" Grid.Row="3" Grid.Column="1">
            <TextBlock Text="Email: " Margin="2"></TextBlock>
            <TextBlock Text="{Binding Email}" Margin="2"></TextBlock>
        </StackPanel>

        </Grid>
    </DataTemplate>
</Window.Resources>
<Grid x:Name="gridClientes">
    <Grid.RowDefinitions>
        <RowDefinition Height="32*" />
        <RowDefinition Height="229*" />
    </Grid.RowDefinitions>
    <TextBlock Height="23" HorizontalAlignment="Left" Margin="65,4,0,0"
```

```
            Name="textBlock1"
            Text="Relação dos Clientes Cadastrados"
            VerticalAlignment="Top"
            TextAlignment="Center"
            FontSize="16"
            FontFamily="Segoe UI Mono"
            FontStretch="Expanded"
            FontWeight="Bold"
            ForceCursor="True"
            Width="317">
        <TextBlock.Background>
          <LinearGradientBrush EndPoint="0.5,1" StartPoint="0.5,0">
            <GradientStop Color="#FFEBBEBE" Offset="0.07" />
            <GradientStop Color="#FFDEB6B6" Offset="1" />
            <GradientStop Color="#FFC77676" Offset="0.715" />
          </LinearGradientBrush></TextBlock.Background>
        <TextBlock.Foreground>
          <LinearGradientBrush EndPoint="1,0.5" StartPoint="0,0.5">
            <GradientStop Color="Black" Offset="0" />
            <GradientStop Color="White" Offset="1" />
          </LinearGradientBrush>
        </TextBlock.Foreground>
      </TextBlock>
      <Border Grid.Row="1" Background="#FFE5CCA1">
          <ListBox Grid.Row="1" ItemTemplate="{StaticResource listBoxTemplate}"
Name="listaClientes"
                   ItemsPanel="{StaticResource ItemTemplate}" Width="550"
                   ScrollViewer.HorizontalScrollBarVisibility="Disabled" />
      </Border>
    </Grid>
</Window>
```

C# da janela anterior

```
public partial class jRelacaoClientes : Window
  {
    List<clsCliente> clientes = new List<clsCliente>();

    public jRelacaoClientes()
    {
      InitializeComponent();
    }

    private void Window_Loaded(object sender, RoutedEventArgs e)
    {
      try
      {
        clientes.Clear();
        clsCliente cliente = new clsCliente();

        foreach (clsCliente c in cliente.GetClientes())
        {
          clientes.Add(c);
```

```
            }
            gridClientes.DataContext = clientes;
            listaClientes.ItemsSource = clientes;
        }
        catch (Exception ex)
        {
            MessageBox.Show("Falha na operação." + ex.Message);
        }
    }
}
```

Catálogo de produtos cadastrados

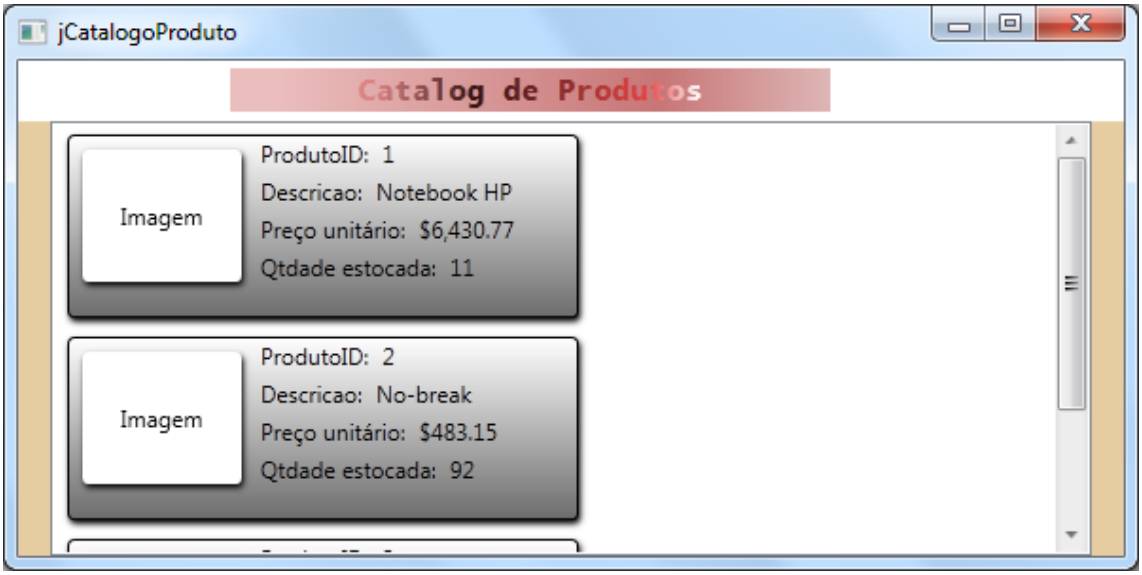

Figura 9: Catálogo de produtos cadastrados

XAML do relatório Catálogo de Produtos

```
<Window x:Class="SistemaPontoVenda.jCatalogoProduto"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="jCatalogoProduto" Height="300" Width="600"
WindowStartupLocation="CenterOwner"
      Loaded="Window_Loaded">
   <Window.Resources>
      <ItemsPanelTemplate x:Key="ItemTemplate">
         <WrapPanel />
      </ItemsPanelTemplate>
      <DataTemplate x:Key="CatalogoDeProdutos">
         <Grid Width="270" Margin="5">
            <Grid.ColumnDefinitions>
               <ColumnDefinition Width="100" />
               <ColumnDefinition />
            </Grid.ColumnDefinitions>
            <Grid.RowDefinitions>
```

```
                <RowDefinition Height="Auto"></RowDefinition>
                <RowDefinition Height="Auto"></RowDefinition>
                <RowDefinition Height="Auto"></RowDefinition>
                <RowDefinition Height="Auto"></RowDefinition>
                <RowDefinition Height="Auto"></RowDefinition>
            </Grid.RowDefinitions>
            <Rectangle Stroke="Black" RadiusX="3" RadiusY="3" Grid.RowSpan="5"
Grid.ColumnSpan="2">
                <Rectangle.Fill>
                    <LinearGradientBrush StartPoint="0,0" EndPoint="0,1">
                        <GradientStop Color="White" Offset="0"/>
                        <GradientStop Color="#6D6D6D" Offset="1"/>
                    </LinearGradientBrush>
                </Rectangle.Fill>
                <Rectangle.Effect>
                    <DropShadowEffect ShadowDepth="2"/>
                </Rectangle.Effect>
            </Rectangle>
            <Border BorderBrush="Black" Background="White" Margin="8" Height="70"
                    VerticalAlignment="Top" CornerRadius="3" Grid.RowSpan="5">
                <Border.Effect>
                    <DropShadowEffect ShadowDepth="2"/>
                </Border.Effect>
                <TextBlock Text="Imagem" VerticalAlignment="Center"
HorizontalAlignment="Center" />
            </Border>
            <StackPanel Orientation="Horizontal" Grid.Row="0" Grid.Column="1">
                <TextBlock Text="ProdutoID: " Margin="2"/>
                <TextBlock Text="{Binding ProdutoID}" Margin="2"></TextBlock>
            </StackPanel>
            <StackPanel Orientation="Horizontal" Grid.Row="1" Grid.Column="1">
                <TextBlock Text="Descricao: " Margin="2"></TextBlock>
                <TextBlock Text="{Binding Descricao}" Margin="2"></TextBlock>
            </StackPanel>
            <StackPanel Orientation="Horizontal" Grid.Row="2" Grid.Column="1">
                <TextBlock Text="Preço unitário: " Margin="2"></TextBlock>
                <TextBlock Text="{Binding PrecoUnitario, StringFormat=C}"
Margin="2"></TextBlock>
            </StackPanel>
            <StackPanel Orientation="Horizontal" Grid.Row="3" Grid.Column="1">
                <TextBlock Text="Qtdade estocada: " Margin="2"></TextBlock>
                <TextBlock Text="{Binding Estocada}" Margin="2"></TextBlock>
            </StackPanel>

        </Grid>
    </DataTemplate>
  </Window.Resources>
  <Grid x:Name="gridCatalog">
    <Grid.RowDefinitions>
      <RowDefinition Height="32*" />
      <RowDefinition Height="229*" />
    </Grid.RowDefinitions>
    <TextBlock Height="23" HorizontalAlignment="Left" Margin="112,4,0,0"
Name="textBlock1"
```

```
            Text="Catalog de Produtos" VerticalAlignment="Top" TextAlignment="Center"
FontSize="16" FontFamily="Segoe UI Mono" FontStretch="Expanded" FontWeight="Bold"
ForceCursor="True" Width="317"><TextBlock.Background><LinearGradientBrush
EndPoint="1,0.5" StartPoint="0,0.5"><GradientStop Color="#FFEBBEBE" Offset="0.07"
/><GradientStop Color="#FFDEB6B6" Offset="1" /><GradientStop Color="#FFC77676"
Offset="0.715"
/></LinearGradientBrush></TextBlock.Background><TextBlock.Foreground><LinearGradient
Brush EndPoint="1,0.5" StartPoint="0,0.5"><GradientStop Color="#FFDE8383"
Offset="0.047" /><GradientStop Color="White" Offset="1" /><GradientStop
Color="#FF341212" Offset="0.343" /><GradientStop Color="#FFE54343" Offset="0.843"
/></LinearGradientBrush></TextBlock.Foreground></TextBlock>
      <Border Grid.Row="1" Background="#FFE5CCA1">
         <ListBox Grid.Row="1" ItemTemplate="{StaticResource CatalogoDeProdutos}"
Name="lstCatalogProduto"
                  ItemsPanel="{StaticResource ItemTemplate}" Width="550"
                  ScrollViewer.HorizontalScrollBarVisibility="Disabled" />
      </Border>
   </Grid>
</Window>
```

C# do relatório Catálogo de produtos

```
public partial class jCatalogoProduto : Window
   {
      public jCatalogoProduto()
      {
         InitializeComponent();
      }

      private void Window_Loaded(object sender, RoutedEventArgs e)
      {
         Produto produto = new Produto();
         lstCatalogProduto.ItemsSource = produto.RetornarTodosProdutos();
      }
   }
```

Relatório de vendas realizadas

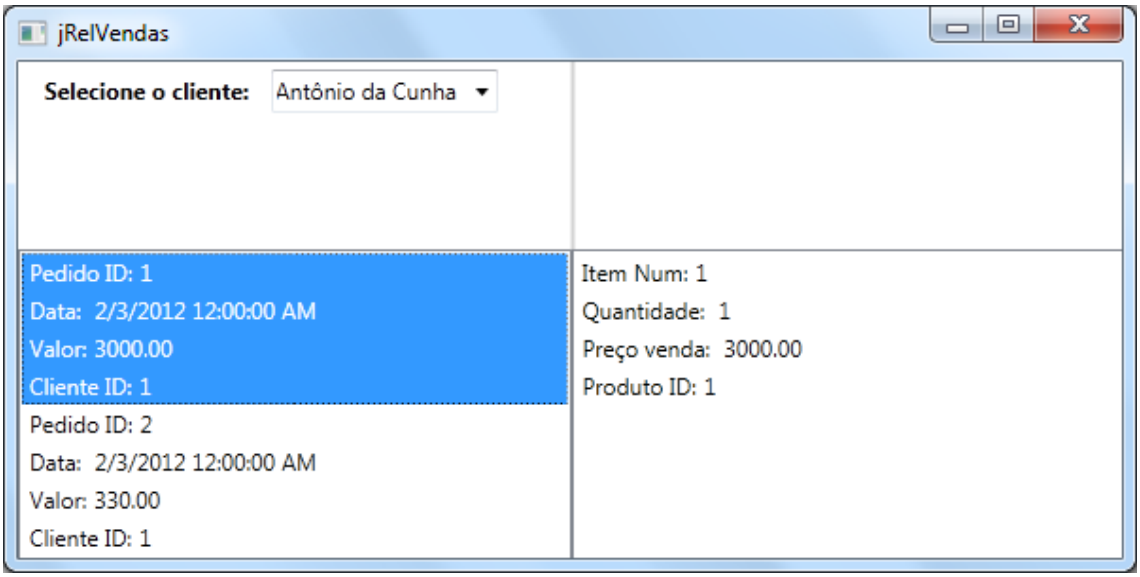

Figura 10: Relatório de vendas realizadas a cada cliente

XAML do relatório Vendas realizadas

```
<Window x:Class="SistemaPontoVenda.jRelVendas"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      Title="jRelVendas" Height="300" Width="600" Loaded="Window_Loaded"
      WindowStartupLocation="CenterOwner">
   <Window.Resources>
      <DataTemplate x:Key="lstPedidos">
         <Grid>
            <Grid.RowDefinitions>
               <RowDefinition Height="Auto"/>
               <RowDefinition Height="Auto"/>
               <RowDefinition Height="Auto"/>
               <RowDefinition Height="Auto"/>
            </Grid.RowDefinitions>
            <StackPanel Orientation="Horizontal" Grid.Row="0"  >
               <TextBlock Text="Pedido ID:" Margin="2"/>
               <TextBlock Text="{Binding PedidoID}" Margin="2"></TextBlock>
            </StackPanel>
            <StackPanel Orientation="Horizontal" Grid.Row="1"  >
               <TextBlock Text="Data: " Margin="2"/>
               <TextBlock Text="{Binding Data}" Margin="2"></TextBlock>
            </StackPanel>
            <StackPanel Orientation="Horizontal" Grid.Row="2"  >
               <TextBlock Text="Valor:" Margin="2"/>
               <TextBlock Text="{Binding Valor}" Margin="2"></TextBlock>
            </StackPanel>
            <StackPanel Orientation="Horizontal" Grid.Row="3"  >
               <TextBlock Text="Cliente ID:" Margin="2"/>
               <TextBlock Text="{Binding ClienteID}" Margin="2"></TextBlock>
            </StackPanel>
         </Grid>
```

```
        </DataTemplate>
        <DataTemplate x:Key="lstItensPedido">
          <Grid>
            <Grid.RowDefinitions>
              <RowDefinition Height="Auto"/>
              <RowDefinition Height="Auto"/>
              <RowDefinition Height="Auto"/>
              <RowDefinition Height="Auto"/>
            </Grid.RowDefinitions>
            <StackPanel Orientation="Horizontal" Grid.Row="0" >
              <TextBlock Text="Item Num:" Margin="2"/>
              <TextBlock Text="{Binding ItemNum}" Margin="2"></TextBlock>
            </StackPanel>
            <StackPanel Orientation="Horizontal" Grid.Row="1"  >
              <TextBlock Text="Quantidade: " Margin="2"/>
              <TextBlock Text="{Binding Qtdade}" Margin="2"></TextBlock>
            </StackPanel>
            <StackPanel Orientation="Horizontal" Grid.Row="2"  >
              <TextBlock Text="Preço venda: " Margin="2"/>
              <TextBlock Text="{Binding PrecoVenda}" Margin="2"></TextBlock>
            </StackPanel>
            <StackPanel Orientation="Horizontal" Grid.Row="3"  >
              <TextBlock Text="Produto ID:" Margin="2"/>
              <TextBlock Text="{Binding ProdutoID}" Margin="2"></TextBlock>
            </StackPanel>
          </Grid>
        </DataTemplate>
      </Window.Resources>
      <Grid Height="263" Width="585">
        <Grid.RowDefinitions>
          <RowDefinition Height="30*" />
          <RowDefinition Height="Auto" />
        </Grid.RowDefinitions>
        <Grid.ColumnDefinitions>
          <ColumnDefinition />
          <ColumnDefinition />
        </Grid.ColumnDefinitions>

        <TextBlock  Height="23" HorizontalAlignment="Left" Margin="14,7,0,0"
Name="textBlock1"
                    Text="Selecione o cliente:"
                    VerticalAlignment="Top" FontWeight="Bold" />
        <ComboBox Height="23" HorizontalAlignment="Left" Margin="134,4,0,0"
                  Name="cboClientes" VerticalAlignment="Top" Width="120"
                  IsEditable="True" DisplayMemberPath="Nome"
                  SelectionChanged="cboClientes_SelectionChanged" />
        <ListBox x:Name="listaPedidos" Grid.Row="1" Height="Auto"
ItemTemplate="{StaticResource lstPedidos}"
                 SelectionChanged="listaPedidos_SelectionChanged" Focusable="False"></ListBox>
        <GridSplitter Name="gspJanela" Grid.Column="1" HorizontalAlignment="Left"
                      VerticalAlignment="Stretch"
                      Width="2" Background="LightGray"/>

          <ListBox x:Name="listaItensPedido" Grid.Row="1" Grid.Column="1" Height="Auto"
                    ItemTemplate="{StaticResource lstItensPedido}"></ListBox>
```

```
    </Grid>
</Window>
```

C# do relatório realizar vendas

```
public partial class jRelVendas : Window
    {
        private Pedido objpedido;

        public jRelVendas()
        {
            InitializeComponent();
        }

        private void cboClientes_SelectionChanged(object sender, SelectionChangedEventArgs e)
        {
            if (cboClientes.SelectedItem != null)
            {
                listaItensPedido.ItemsSource = null;
            }
            clsCliente cliente = new clsCliente();

            cliente = (clsCliente)cboClientes.SelectedItem;
            cliente.ClienteID = cliente.ClienteID ;

            Pedido pedido = new Pedido();

            listaPedidos.ItemsSource = pedido.GetPedidosPorClienteID(cliente.ClienteID);
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            clsCliente cliente = new clsCliente();
            cboClientes.ItemsSource = cliente.GetClientes();
        }

        private void listaPedidos_SelectionChanged(object sender, SelectionChangedEventArgs e)
        {
            objpedido = new Pedido();
            ItensPedido item = new ItensPedido();
            if (listaPedidos.SelectedItem != null)
            {
                objpedido = (Pedido)listaPedidos.SelectedItem;
                listaItensPedido.ItemsSource = item.GetItensPorPedidoID(objpedido.PedidoID);
            }
        }
    }
```