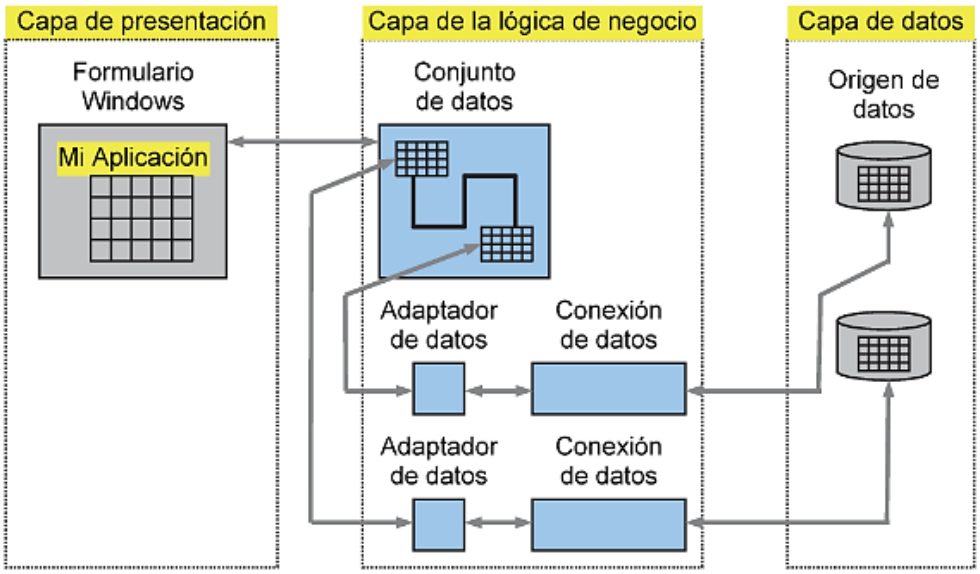
# Semana 13




Crear la base de datos **BDVentas**

**Producto**

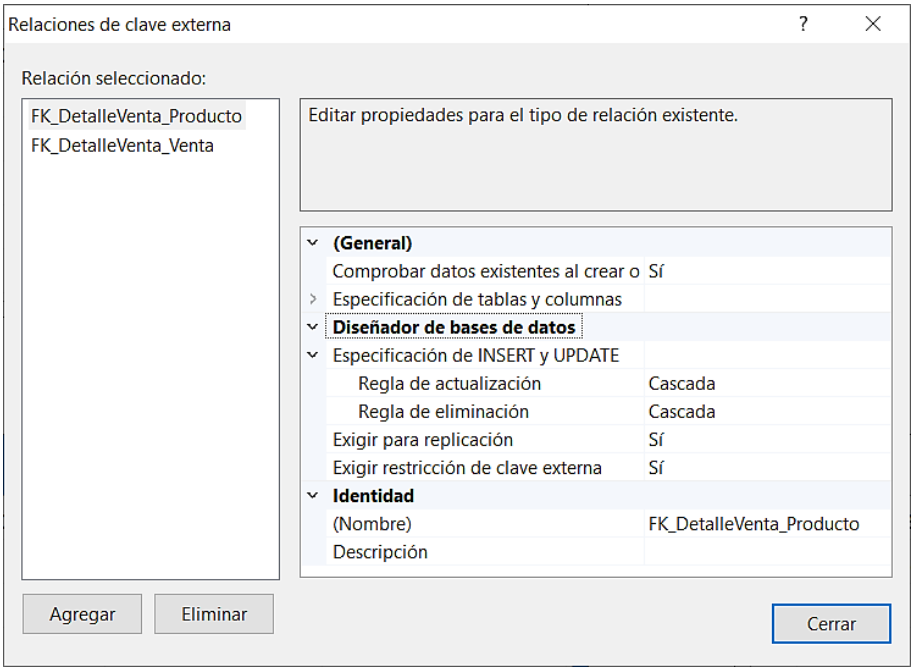
| | Nombre de columna | Tipo de datos | Permitir valores ... |
|---|---|---|---|
| 🔑 | codproducto | int | ☐ |
| | nombre | varchar(100) | ☑ |
| | precio | decimal(18, 2) | ☑ |
| | stock | int | ☑ |
| | | | ☐ |

**DetalleVenta**

| | Nombre de columna | Tipo de datos | Permitir valores ... |
|---|---|---|---|
| 🔑 | codventa | int | ☐ |
| 🔑 | codproducto | int | ☐ |
| | cantidad | int | ☐ |
| | precio | decimal(18, 2) | ☐ |
| | descuento | decimal(18, 2) | ☐ |
| | | | ☐ |

**Venta**

| | Nombre de columna | Tipo de datos | Permitir valores ... |
|---|---|---|---|
| 🔑 | codventa | int | ☐ |
| | cliente | varchar(100) | ☐ |
| | fecha | datetime | ☑ |
| | | | ☐ |



## Crear la Capa de Datos

- Crear nuevo proyecto (CapaPresentacion ➔ **ProySistemaVentas**)
- Instalar el paquete de **System.Data.SqlClient 4.8.3**
- Agregar nuevo proyecto de biblioteca de clases WPF (**CapaDatos**)
- Agregar la clase Conexión

```
class Conexion
    {
        public static string cn = "Data Source=.;Initial Catalog=BDVentas;Integrated
Security=True";
    }
```

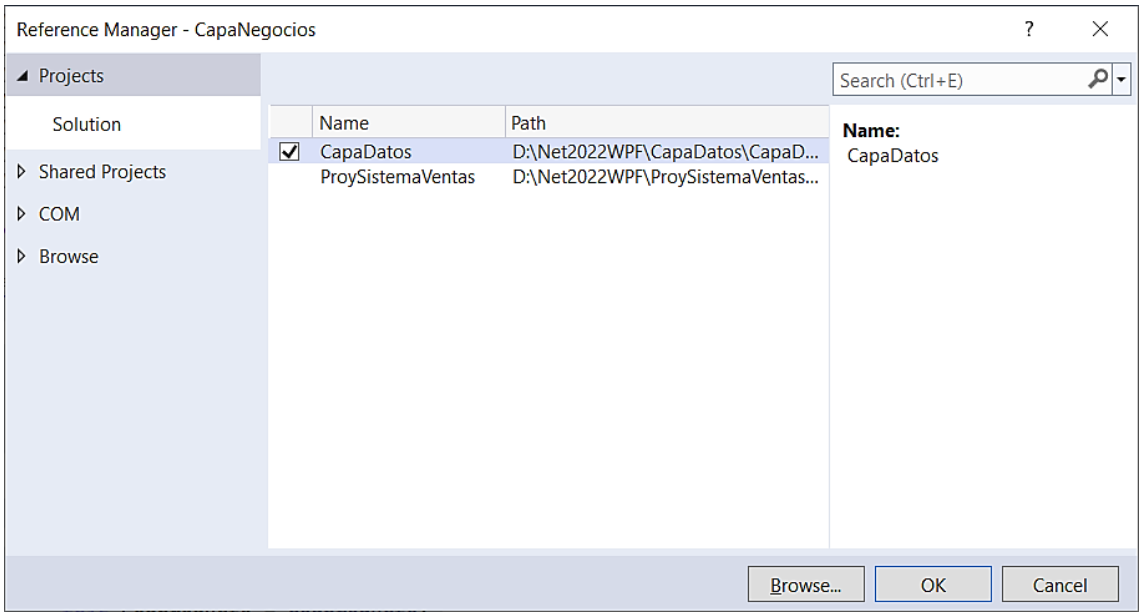
- Agregar la clase Producto

```
public class Productos
{
    private int codproducto;
    private string nombre;
    private decimal precio;
    private int stock;

    public Productos() { }
    public Productos(int codproducto, string nombre, decimal precio, int stock)
    {
        this.Codproducto = codproducto;
        this.Nombre = nombre;
        this.Precio = precio;
        this.Stock = stock;
    }

    public int Codproducto { get => codproducto; set => codproducto = value; }
    public string Nombre { get => nombre; set => nombre = value; }
    public decimal Precio { get => precio; set => precio = value; }
    public int Stock { get => stock; set => stock = value; }

    public string insertar(Productos oProducto) {
        string rpta = "";
        try
        {
            SqlConnection con = new SqlConnection();
            con.ConnectionString = Conexion.cn;
            SqlCommand cmd = new SqlCommand("INSERT INTO Producto VALUES (@nom, @precio, @stock)",
con);
            cmd.Parameters.AddWithValue("@nom", oProducto.nombre);
            cmd.Parameters.AddWithValue("@precio", oProducto.Precio);
            cmd.Parameters.AddWithValue("@stock", oProducto.stock);
            con.Open();
            cmd.ExecuteNonQuery();
            con.Close();
            rpta = "Información registrada";
        }
        catch (SqlException ex)
        {
            rpta = ex.Message;
        }

        return rpta;
    }

    public string actualizar(Productos oProducto) {
        string rpta = "";
        try
        {
            SqlConnection con = new SqlConnection();
            con.ConnectionString = Conexion.cn;
            SqlCommand cmd = new SqlCommand("UPDATE Producto SET nombre = @nom, precio = @precio,
stock = @stock WHERE codproducto = @cod", con);
            cmd.Parameters.AddWithValue("@cod", oProducto.codproducto);
            cmd.Parameters.AddWithValue("@nom", oProducto.nombre);
            cmd.Parameters.AddWithValue("@precio", oProducto.Precio);
```

```
            cmd.Parameters.AddWithValue("@stock", oProducto.stock);
            con.Open();
            cmd.ExecuteNonQuery();
            con.Close();
            rpta = "Registro actualizado";
        }
        catch (SqlException ex)
        {
            rpta= ex.Message;
        }

        return rpta;
    }
    /*public DataTable obtenerProducto() {
        string rpta = "";
        DataTable dtProducto = new DataTable("Productos");

        return dtProducto;
    }*/
}
```

### Crear la Capa Negocios

- Agregar nuevo proyecto de biblioteca de clases WPF (**CapaNegocios**)
- Agregar la referencia hacia la capa de datos

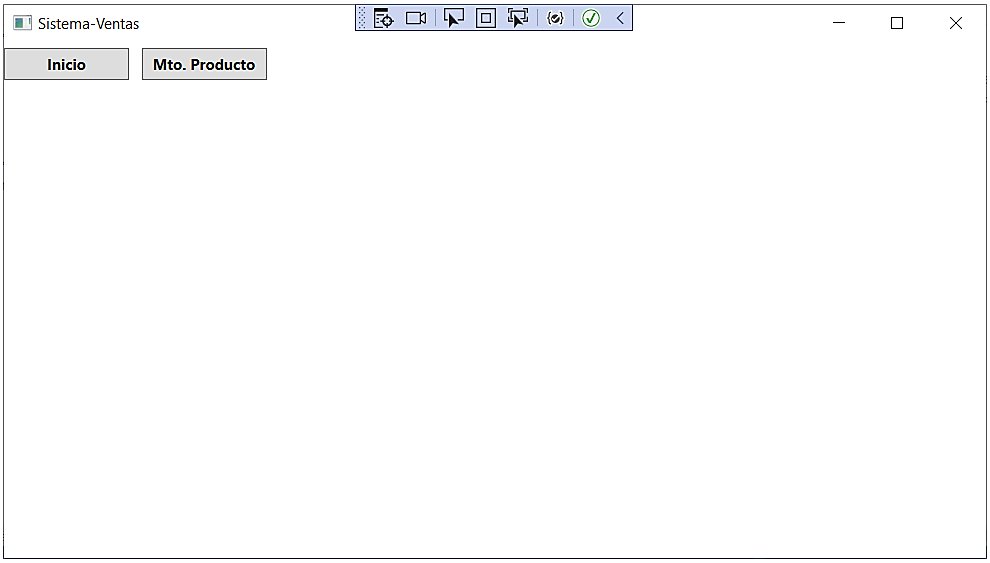



- Agregar la clase NegProducto

```
public class NegProducto
{
    public static string Insertar(string nombre, decimal precio, int stock) {
        Productos pro = new Productos();
        pro.Nombre = nombre;
        pro.Precio = precio;
        pro.Stock = stock;
        return pro.insertar(pro);
    }
    public static string Actualizar(int cod, string nombre, decimal precio, int stock) {
        Productos pro = new Productos();
        pro.Codproducto= cod;
        pro.Nombre = nombre;
        pro.Precio = precio;
        pro.Stock = stock;
        return pro.actualizar(pro);
    }
    /*public static DataTable ObtenerProducto() {
        return new Productos().obtenerProducto();
    }*/
}
```

## Crear la Capa Presentación (ProySistemaVentas)

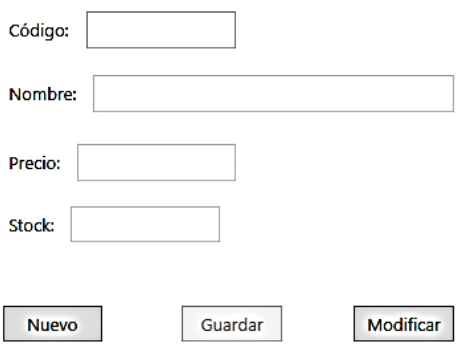
- Configurar la ventana **MainWindow**




```
<Grid>
<StackPanel Orientation="Horizontal" Height="35" VerticalAlignment="Top">
     <Button x:Name="btInicio" Content="Inicio" Width="100" Click="btInicio_Click" Height="25"
FontWeight="Bold"/>
     <Button x:Name="btProducto" Content="Mto. Producto" Width="100" Click="btProducto_Click"
Height="25" FontWeight="Bold" Margin="10,0,0,0"/>
</StackPanel>
<Frame x:Name="Main" Margin="0,35,0,0" NavigationUIVisibility="Hidden" />
</Grid>
```

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

    private void btProducto_Click(object sender, RoutedEventArgs e)
    {
        Main.Content = new ManteProducto();
    }

    private void btInicio_Click(object sender, RoutedEventArgs e)
    {
        Main.Content = new Inicio();
    }
}
```

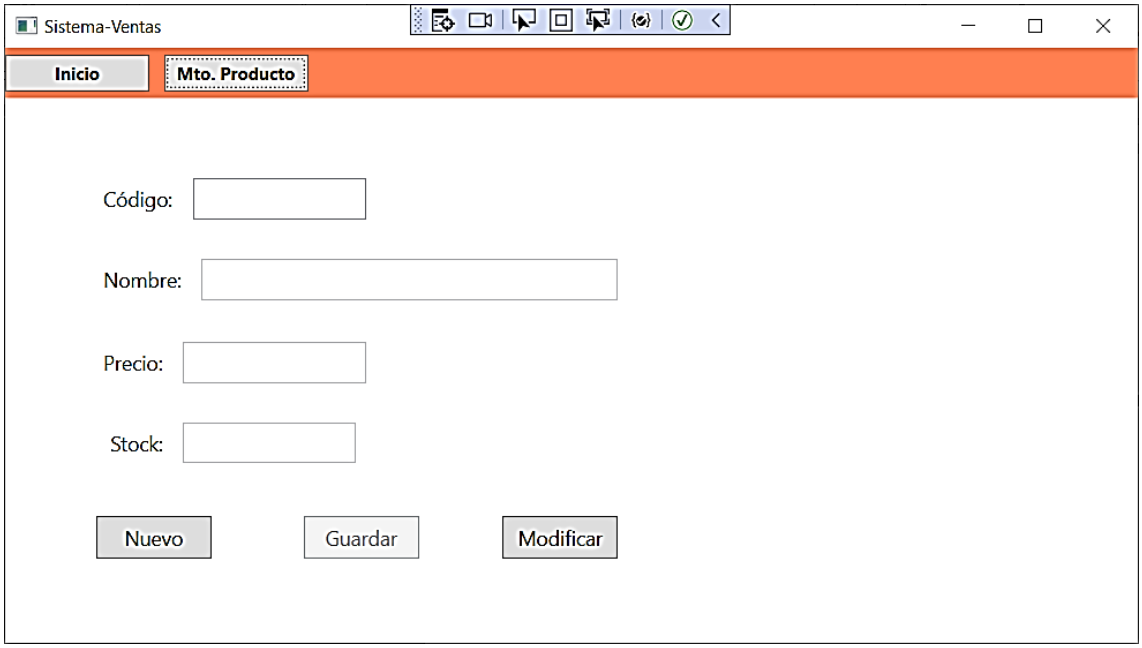
- Agregar una página WPF **ManteProducto**



```
<Grid Background="White">
<Label x:Name="label" Content="Código: " HorizontalAlignment="Left" Margin="63,55,0,0"
VerticalAlignment="Top" FontSize="14"/>
<TextBox x:Name="tbCodigo" HorizontalAlignment="Left" Margin="130,55,0,0" TextWrapping="Wrap"
VerticalAlignment="Top" Width="120" FontSize="14" Height="29"/>
<Label x:Name="label_Copy" Content="Nombre: " HorizontalAlignment="Left" Margin="63,106,0,0"
VerticalAlignment="Top" FontSize="14"/>
<TextBox x:Name="tbNombre" HorizontalAlignment="Left" Margin="136,106,0,0" TextWrapping="Wrap"
VerticalAlignment="Top" Width="289" FontSize="14" Height="29" IsEnabled="False"/>
<Label x:Name="label_Copy1" Content="Precio: " HorizontalAlignment="Left" Margin="63,161,0,0"
VerticalAlignment="Top" FontSize="14"/>
<TextBox x:Name="tbPrecio" HorizontalAlignment="Left" Margin="123,161,0,0" TextWrapping="Wrap"
VerticalAlignment="Top" Width="127" FontSize="14" Height="29" IsEnabled="False"/>
<Label x:Name="label_Copy2" Content="Stock: " HorizontalAlignment="Left" Margin="63,0,0,0"
VerticalAlignment="Center" FontSize="14"/>
<TextBox x:Name="tbStock" HorizontalAlignment="Left" Margin="118,0,0,0" TextWrapping="Wrap"
VerticalAlignment="Center" Width="120" FontSize="14" Height="29" IsEnabled="False"/>
<Button x:Name="btGuardar" Content="Guardar" HorizontalAlignment="Left" Margin="207,290,0,0"
VerticalAlignment="Top" FontSize="14" Width="80" Height="29" Click="btGuardar_Click"
IsEnabled="False"/>
<Button x:Name="btModificar" Content="Modificar" HorizontalAlignment="Left" Margin="345,290,0,0"
VerticalAlignment="Top" FontSize="14" Width="80" Height="29" Click="btModificar_Click"/>
<Button x:Name="btNuevo" Content="Nuevo" HorizontalAlignment="Left" Margin="63,290,0,0"
VerticalAlignment="Top" FontSize="14" Width="80" Height="29" Click="btNuevo_Click"/>
</Grid>
```

```
public partial class ManteProducto : Page
{
    public ManteProducto()
    {
        InitializeComponent();
    }
    private void btGuardar_Click(object sender, RoutedEventArgs e)
    {
        if (tbCodigo.Text == "")
        {
            string rpta = "";
            rpta = NegProducto.Insertar(tbNombre.Text.Trim().ToUpper(),
Convert.ToDecimal(tbPrecio.Text), Convert.ToInt32(tbStock.Text));
            MessageBox.Show(rpta, "Seguridad", MessageBoxButton.OK, MessageBoxImage.Information);
            //MessageBox.Show("Se guarda nuevo registro");
        }
        else
        {
            string rpta = "";
            rpta = NegProducto.Actualizar(Convert.ToInt32(tbCodigo.Text),
tbNombre.Text.Trim().ToUpper(), Convert.ToDecimal(tbPrecio.Text), Convert.ToInt32(tbStock.Text));
            MessageBox.Show(rpta, "Seguridad", MessageBoxButton.OK, MessageBoxImage.Information);
            //MessageBox.Show("Se actualiza nuevo registro");
        }
        tbNombre.IsEnabled = false;
        tbPrecio.IsEnabled = false;
        tbStock.IsEnabled = false;
        btGuardar.IsEnabled = false;
        tbCodigo.IsEnabled = true;
        tbCodigo.Focus();
        btModificar.IsEnabled = true;

        tbCodigo.Clear();
        tbNombre.Clear();
        tbPrecio.Clear();
        tbStock.Clear();
    }

    private void btModificar_Click(object sender, RoutedEventArgs e)
    {
        if (tbCodigo.Text == "")
        {
            MessageBox.Show("Indicar el codigo del producto a modificar");
            tbCodigo.Focus();
        }
```

```
        else
        {
            tbNombre.IsEnabled = true;
            tbNombre.Focus();
            tbPrecio.IsEnabled = true;
            tbStock.IsEnabled = true;
            btGuardar.IsEnabled = true;
            tbCodigo.IsEnabled = false;
            btModificar.IsEnabled = false;
        }
    }
    private void btNuevo_Click(object sender, RoutedEventArgs e)
    {
        tbNombre.IsEnabled = true;
        tbNombre.Focus();
        tbPrecio.IsEnabled = true;
        tbStock.IsEnabled = true;
        btGuardar.IsEnabled = true;
        tbCodigo.IsEnabled = false;
        btModificar.IsEnabled = false;
    }
}
```



Implemetar el metodo **obtenerProducto** en la clase **Productos**

```
public DataTable obtenerProducto() {
    string rpta = "";
    DataTable dtProducto = new DataTable("Productos");

    try
    {
        SqlConnection con = new SqlConnection();
        SqlDataAdapter da = new SqlDataAdapter();
        con.ConnectionString = Conexion.cn;
        SqlCommand cmd = new SqlCommand("SELECT codproducto, nombre, precio, stock FROM Producto",
con);
        da.SelectCommand = cmd;
        da.Fill(dtProducto);
    }
    catch (SqlException ex) {
        rpta = ex.Message;
        dtProducto = null;
    }
    return dtProducto;
}
```

Implemetar el metodo **ObtenerProducto** en la clase **NegProductos**

```
public static DataTable ObtenerProducto() {
    return new Productos().obtenerProducto();
}
```

Implementar los siguientes procedimientos almacenados en la **BDVentas**

```
CREATE PROC spI_Venta
   @codigoVenta  int = Null OUTPUT,
   @cliente  varchar(100) = Null
AS
insert into Venta(cliente,fecha)
VALUES(@cliente,GETDATE())
--Obteniendo el codigo autogenerado de la venta
SET @codigoVenta = @@IDENTITY


CREATE PROC spI_DetalleVenta
   @codigoVenta  int = Null,
   @codigoProducto  int = Null,
   @cantidad  int = Null,
   @precio  decimal(18, 2) = Null,
   @descuento  decimal(18, 2) = Null
AS
insert into DetalleVenta ( codVenta, codproducto, cantidad,
precio,descuento )
VALUES( @codigoVenta, @codigoProducto, @cantidad,@precio ,@descuento )
```

Implementar las clases DetalleVenta y Venta en **CapaDatos**

**DetalleVenta.cs**

```
public class DetalleVenta
    {
        public int codVenta;
        public int codProducto;
        public int cantidad;
        public decimal precio;
        public decimal descuento;

        public DetalleVenta() { }
        public DetalleVenta(int codVenta, int codProducto, int cantidad, decimal precio, decimal
descuento)
        {
            this.codVenta = codVenta;
            this.codProducto = codProducto;
            this.cantidad = cantidad;
            this.precio = precio;
            this.descuento = descuento;
        }

        public string Insertar(DetalleVenta oDetalleVenta, ref SqlConnection sqlCon, ref
SqlTransaction sqlTra)
        {
            string rpta = "";
            try
            {
                //1. Establecer el comando
                SqlCommand sqlCmd = new SqlCommand();
                sqlCmd.Connection = sqlCon;
                sqlCmd.Transaction = sqlTra;
                sqlCmd.CommandText = "spI_DetalleVenta";
                sqlCmd.CommandType = CommandType.StoredProcedure;
                //4. Agregar los parametros al comando
                //Establecemos los valores para el parametro @codigoVenta delProcedimiento
Almacenado
                SqlParameter sqlParcodigoVenta = new SqlParameter();
                sqlParcodigoVenta.ParameterName = "@codigoVenta";
                sqlParcodigoVenta.SqlDbType = SqlDbType.Int;
                sqlParcodigoVenta.Value = oDetalleVenta.codVenta;
                sqlCmd.Parameters.Add(sqlParcodigoVenta);
                //Agregamos el parametro al comando
                //Establecemos los valores para el parametro @codigoProducto del Procedimiento
Almacenado
                SqlParameter sqlParcodigoProducto = new SqlParameter();
                sqlParcodigoProducto.ParameterName = "@codigoProducto";
```

```
                sqlParcodigoProducto.SqlDbType = SqlDbType.Int;
                sqlParcodigoProducto.Size = 4;
                sqlParcodigoProducto.Value = oDetalleVenta.codProducto;
                sqlCmd.Parameters.Add(sqlParcodigoProducto);
                //Agregamos el parametro al comando
                //Establecemos los valores para el parametro @cantidad del Procedimiento
Almacenado
                SqlParameter sqlParcantidad = new SqlParameter();
                sqlParcantidad.ParameterName = "@cantidad";
                sqlParcantidad.SqlDbType = SqlDbType.Int;
                sqlParcantidad.Size = 4;
                sqlParcantidad.Value = oDetalleVenta.cantidad;
                sqlCmd.Parameters.Add(sqlParcantidad);
                //Agregamos el parametro al comando
                //Establecemos los valores para el parametro @precio del Procedimiento
Almacenado
                SqlParameter sqlParprecio = new SqlParameter();
                sqlParprecio.ParameterName = "@precio";
                sqlParprecio.SqlDbType = SqlDbType.Decimal;
                sqlParprecio.Precision = 18;
                sqlParprecio.Scale = 2;
                sqlParprecio.Value = oDetalleVenta.precio;
                sqlCmd.Parameters.Add(sqlParprecio);
                //Agregamos el parametro al comando
                //Establecemos los valores para el parametro @descuento del Procedimiento
Almacenado
                SqlParameter sqlPardescuento = new SqlParameter();
                sqlPardescuento.ParameterName = "@descuento";
                sqlPardescuento.SqlDbType = SqlDbType.Decimal;
                sqlParcantidad.Precision = 18;
                sqlParcantidad.Scale = 2;
                sqlPardescuento.Value = oDetalleVenta.descuento;
                sqlCmd.Parameters.Add(sqlPardescuento);
                //Agregamos el parametro al comando
                //5. Ejecutamos el commando
                rpta = sqlCmd.ExecuteNonQuery() == 1 ? "OK" : "No se inserto el detalle de venta de
forma correcta";
            }
            catch (Exception ex)
            {
                rpta = ex.Message;
            }
            return rpta;
        }
    }
```

**Venta.cs**

```
public class Venta
    {
        public int codVenta;
        public string cliente;
        public DateTime fecha;

        public Venta() { }
        public Venta(int codVenta, string cliente, DateTime fecha)
        {
            this.codVenta = codVenta;
            this.cliente = cliente;
            this.fecha = fecha;
        }

        public string Insertar(Venta oVenta, List<DetalleVenta> detalles) {
            string rpta = "";
            SqlConnection sqlCon = new SqlConnection();
            try
            {
                //1. Establecer la cadena de conexion
                sqlCon.ConnectionString = Conexion.cn;
                sqlCon.Open();
                //3. Establecer la transaccion
                SqlTransaction sqlTra = sqlCon.BeginTransaction();
                //4. Establecer el comando
```

```
                SqlCommand sqlCmd = new SqlCommand();
                sqlCmd.Connection = sqlCon;
                sqlCmd.Transaction = sqlTra;
                sqlCmd.CommandText = "spI_Venta";
                sqlCmd.CommandType = CommandType.StoredProcedure;


                SqlParameter sqlParcodigoVenta = new SqlParameter();
                sqlParcodigoVenta.ParameterName = "@codigoVenta";
                sqlParcodigoVenta.SqlDbType = SqlDbType.Int;
                sqlParcodigoVenta.Size = 4;
                sqlParcodigoVenta.Direction = ParameterDirection.Output;
                sqlCmd.Parameters.Add(sqlParcodigoVenta); //Agregamos el parametro al comando
                //Establecemos los valores para el parametro @cliente del Procedimiento Almacenado
        SqlParameter sqlParcliente = new SqlParameter();
                sqlParcliente.ParameterName = "@cliente";
                sqlParcliente.SqlDbType = SqlDbType.VarChar;
                sqlParcliente.Size = 100;
                sqlParcliente.Value = oVenta.cliente;
                sqlCmd.Parameters.Add(sqlParcliente); //Agregamos el parametro al comando
                                                      //6. Ejecutamos el commando
                rpta = sqlCmd.ExecuteNonQuery() == 1 ? "OK" : "No se inserto el detalle de venta de
forma correcta";
         if (rpta.Equals("OK"))
                {
                    //Obtenemos el codigo de la venta que se genero por la base de datos
                    this.codVenta = Convert.ToInt32(sqlCmd.Parameters["@codigoVenta"].Value);
                    foreach (DetalleVenta det in detalles)
                    {
                        //Establecemos el codigo de la venta que se autogenero
                        det.codVenta = this.codVenta;
                        //Llamamos al metodo insertar de la clase DetalleVenta
                        //y le pasamos la conexion y la transaccion que debe de usar
                        rpta = det.Insertar(det, ref sqlCon, ref sqlTra);
                        if (!rpta.Equals("OK"))
                        {
                            //Si ocurre un error al insertar un detalle de venta salimos del for
                            break;
                        }
                    }
                }
                if (rpta.Equals("OK"))
                {
                    //Se inserto todo los detalles y confirmamos la transaccion
                    sqlTra.Commit();
                }
                else
                {
                    //Algun detalle no se inserto y negamos la transaccion
                    sqlTra.Rollback();
                }
            }
            catch (Exception ex)
            {
                rpta = ex.Message;
            }
            finally
            {
                //6. Cerramos la conexion con la BD
                if (sqlCon.State == ConnectionState.Open) sqlCon.Close();
            }
            return rpta;
        }

        public DataTable ObtenerVenta(int codigoVenta)
        {
            DataTable dtVenta = new DataTable("Venta");
            SqlConnection sqlCon = new SqlConnection();
            string rpta = "";
            try
            {
                //1. Establecer la cadena de conexion
                sqlCon.ConnectionString = Conexion.cn;
```

```
                //2. Establecer el comando
                SqlCommand sqlCmd = new SqlCommand();
                sqlCmd.Connection = sqlCon;//La conexion que va a usar el comando
                sqlCmd.CommandText = "spS_Venta_Detalle";//El comando a ejecutar
                sqlCmd.CommandType = CommandType.StoredProcedure;
                //Decirle al comando que va a ejecutar una sentencia SQL
                //3. Agregar los parametros al comando
                //Establecemos los valores para el parametro @codigoVenta del Procedimiento
Almacenado
                SqlParameter sqlParcodigoVenta = new SqlParameter();
                sqlParcodigoVenta.ParameterName = "@codigoVenta";
                sqlParcodigoVenta.SqlDbType = SqlDbType.Int;
                sqlParcodigoVenta.Value = codigoVenta;
                sqlCmd.Parameters.Add(sqlParcodigoVenta); //Agregamos el parametro al comando
                //4. El DataAdapter que va a ejecutar el comando y es el encargado de llena el
DataTable
                SqlDataAdapter sqlDat = new SqlDataAdapter(sqlCmd);
                sqlDat.Fill(dtVenta);//Llenamos el DataTable
            }
            catch (Exception ex)
            {
                rpta = ex.Message;
                dtVenta = null;
            }
            return dtVenta;
        }

        public DataTable ObtenerVenta()
        {
            DataTable dtVenta = new DataTable("Venta");
            SqlConnection sqlCon = new SqlConnection();
            string rpta = "";
            try
            {
                //1. Establecer la cadena de conexion
                sqlCon.ConnectionString = Conexion.cn;
                //2. Establecer el comando
                SqlCommand sqlCmd = new SqlCommand();
                sqlCmd.Connection = sqlCon;//La conexion que va a usar el comando
                sqlCmd.CommandText = "spF_Venta_Todos";//El comando a ejecutar
                sqlCmd.CommandType = CommandType.StoredProcedure;//Decirle al comando que va a
ejecutar una sentencia SQL
                //3. No hay parametros
                //4. El DataAdapter que va a ejecutar el comando y es el encargado de llena el
DataTable
                SqlDataAdapter sqlDat = new SqlDataAdapter(sqlCmd);
                sqlDat.Fill(dtVenta);//Llenamos el DataTable
            }
            catch (Exception ex)
            {
                rpta = ex.Message;
                dtVenta = null;
            }
            return dtVenta;
        }
    }
```

Implementar las clases NegDetalleVenta y NegVenta en **CapaNegocios**, no olvidar la dependencia hacia la CapaDatos.

**NegDetalleVenta.cs**

```
public class NegDetalleVenta
    {
        public static decimal ObtenerDescuento(decimal cantidad, decimal pu)
        {
            if ((cantidad * pu) > 50)
            {
                decimal porcentaje = Convert.ToDecimal(0.05);
                decimal descuento = ((cantidad * pu) * porcentaje);
                return descuento;
            }
```

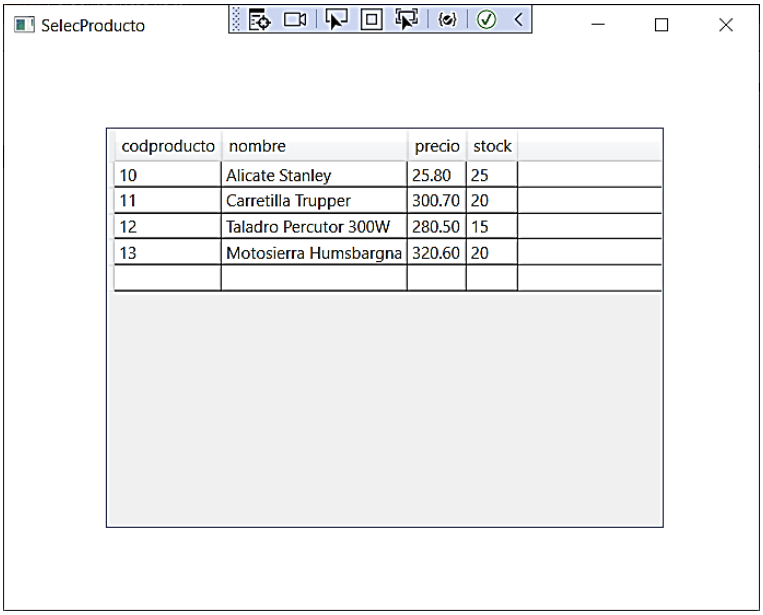
```
            else
            {
                return 0;
            }
        }
    }
```

**NegVenta.cs**

```
public class NegVenta
    {
        public static string Insertar(string cliente, DataTable dtDetalles)
        {
            Venta venta = new Venta();
            venta.cliente = cliente;
            List<DetalleVenta> odetalles = new List<DetalleVenta>();
            foreach (DataRow row in dtDetalles.Rows)
            {
                DetalleVenta detalle = new DetalleVenta();
                detalle.codProducto = Convert.ToInt32(row["codproducto"].ToString());
                detalle.cantidad = Convert.ToInt32(row["cantidad"].ToString());
                detalle.precio = Convert.ToDecimal(row["precio"].ToString());
                detalle.descuento = NegDetalleVenta.ObtenerDescuento(detalle.cantidad,
        Convert.ToDecimal(row["precio"].ToString()));
                odetalles.Add(detalle);
            }
            return venta.Insertar(venta, odetalles);
        }
        //Metodo que se encarga de llamar al metodo ObtenerProducto
        //de la clase Venta
        public static DataTable ObtenerVenta()
        {
            return new Venta().ObtenerVenta();
        }
        //Metodo que se encarga de llamar al metodo ObtenerProducto
        //por codigo de la clase Venta
        public static DataTable ObtenerVenta(int codigoVenta)
        {
            return new Venta().ObtenerVenta(codigoVenta);
        }
    }
```

Implementar la ventana SelecProducto y la pagina VentaDetalle en la CapaPresentación (**ProySistemaVentas**); no olvidar las dependencias hacia CapaDatos y CapaNegocios

**SelecProducto.xaml** (Window WPF)

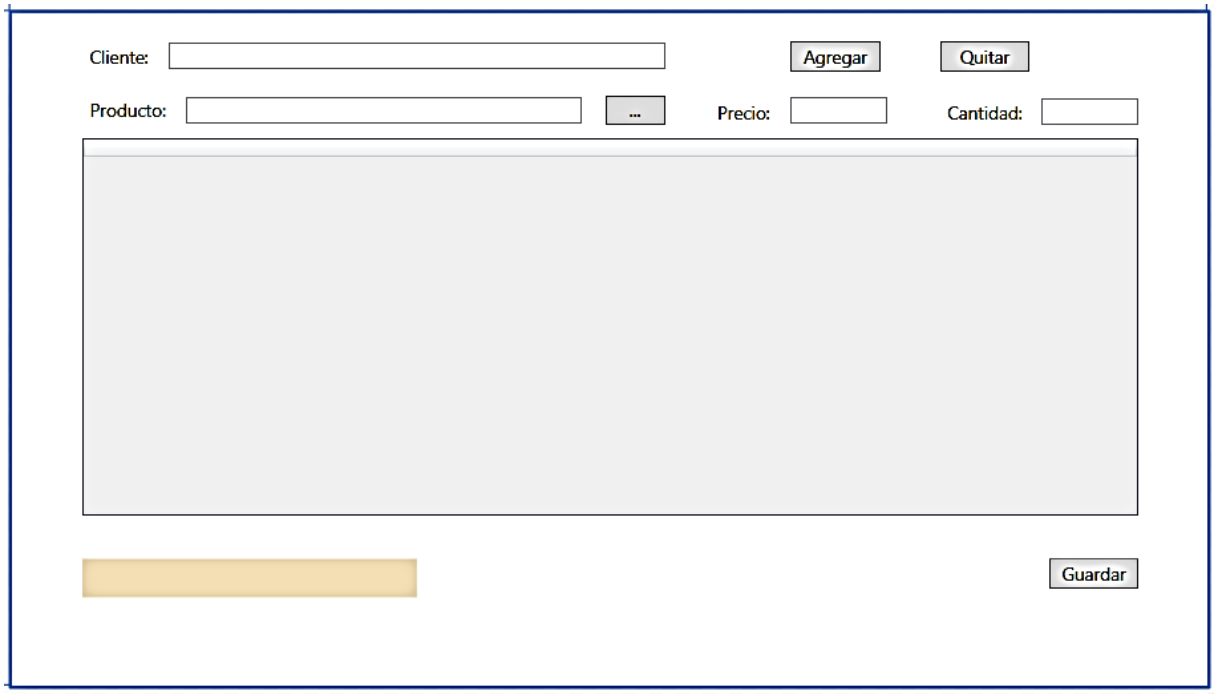


```
<Grid>
    <DataGrid x:Name="dgProducto" Margin="74,60,70,60" ItemsSource="{Binding}"
SelectionChanged="dgProducto_SelectionChanged"/>
</Grid>
```

### SelecProducto.xaml.cs

```
public partial class SelecProducto : Window
    {
        private VentaDetalles frame;

        public SelecProducto()
        {
            InitializeComponent();
        }

        public void estableceVentana(VentaDetalles frame) {
            this.frame = frame;
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            this.dgProducto.AutoGenerateColumns = true;
            this.dgProducto.DataContext = NegProducto.ObtenerProducto();
        }

        private void dgProducto_SelectionChanged(object sender, SelectionChangedEventArgs e)
        {
            DataRowView view = (DataRowView)dgProducto.SelectedItem;
            this.frame.codigoProductoSeleccionado = Convert.ToInt32(view.Row.ItemArray[0]);
            this.frame.tbProduc.Text = Convert.ToString(view.Row.ItemArray[1]);
            this.frame.tbPrecio.Text = Convert.ToString(view.Row.ItemArray[2]);
            this.Hide();
        }
    }
```

### VentaDetalles.xaml (Page WPF)




```
<Grid Background="White">
        <Label x:Name="label" Content="Cliente: " HorizontalAlignment="Left" Margin="47,16,0,0"
VerticalAlignment="Top"/>
        <TextBox x:Name="tbCliente" HorizontalAlignment="Left" Margin="105,20,0,0"
TextWrapping="Wrap" VerticalAlignment="Top" Width="332"/>
        <Label x:Name="label_Copy" Content="Producto: " HorizontalAlignment="Left"
Margin="47,52,0,0" VerticalAlignment="Top"/>
```

```
        <TextBox x:Name="tbProduc" HorizontalAlignment="Left" Margin="116,56,0,0"
TextWrapping="Wrap" VerticalAlignment="Top" Width="265"/>
        <Button x:Name="btBuscar" Content="..." HorizontalAlignment="Left" Margin="397,55,0,0"
VerticalAlignment="Top" RenderTransformOrigin="0.512,0.053" Width="40" Click="btBuscar_Click"/>
        <Label x:Name="label_Copy1" Content="Precio: " HorizontalAlignment="Left"
Margin="467,53,0,0" VerticalAlignment="Top"/>
        <TextBox x:Name="tbPrecio" HorizontalAlignment="Left" Margin="521,56,0,0"
TextWrapping="Wrap" VerticalAlignment="Top" Width="65"/>
        <Label x:Name="label_Copy2" Content="Cantidad: " HorizontalAlignment="Left"
Margin="621,53,0,0" VerticalAlignment="Top"/>
        <TextBox x:Name="tbCant" HorizontalAlignment="Left" Margin="689,57,0,0" TextWrapping="Wrap"
VerticalAlignment="Top" Width="65"/>
        <Button x:Name="btAgregar" Content="Agregar" HorizontalAlignment="Left" Margin="521,19,0,0"
VerticalAlignment="Top" Width="60" Click="btAgregar_Click"/>
        <Button x:Name="btQuitar" Content="Quitar" HorizontalAlignment="Left" Margin="621,19,0,0"
VerticalAlignment="Top" Width="60" Click="btQuitar_Click"/>
        <DataGrid x:Name="dgDetalle" Margin="47,84,46,114" ItemsSource="{Binding}" />
        <Button x:Name="tbGuardar" Content="Guardar" HorizontalAlignment="Left"
Margin="694,365,0,0" VerticalAlignment="Top" Width="60" Click="tbGuardar_Click"/>
        <Label x:Name="lbTotalPagar" Content="" HorizontalAlignment="Left" Margin="47,365,0,0"
VerticalAlignment="Top" Width="224" Background="#FFF3DFB3"/>
</Grid>
```

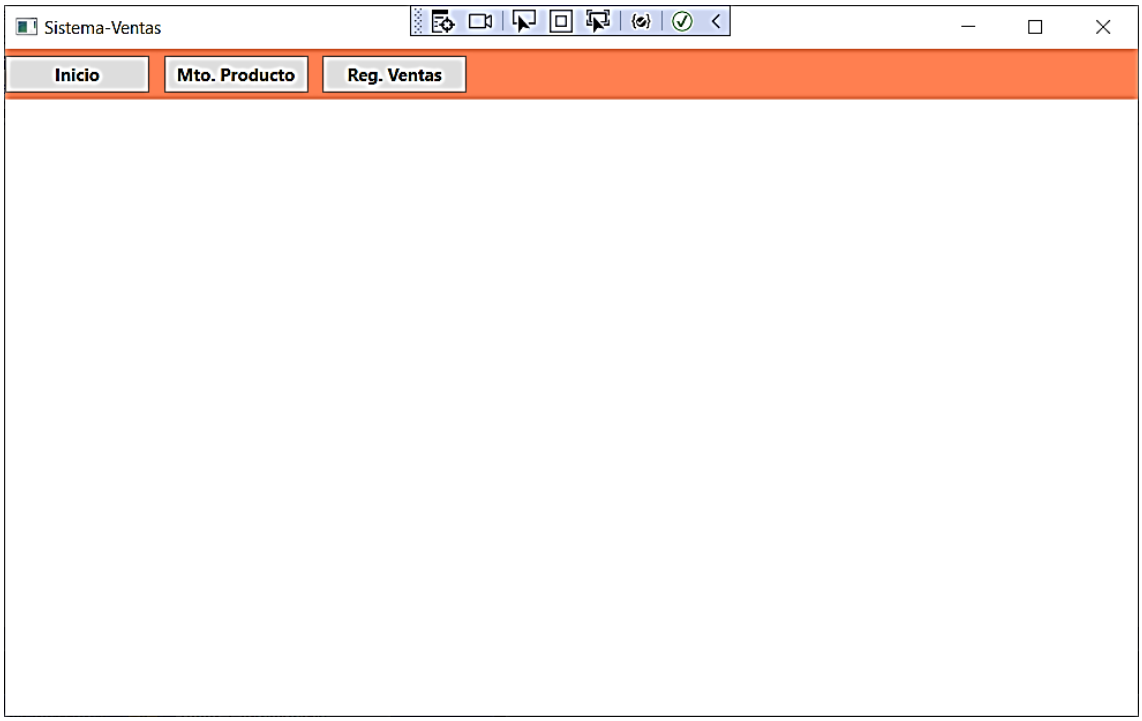
**VentaDetalles.xaml.cs**

```
public partial class VentaDetalles : Page
    {
        private DataTable dtDetalle;
        internal int codigoProductoSeleccionado = -1;
        private decimal totalPagar = 0;
        public VentaDetalles()
        {
            InitializeComponent();
        }

        private void limpiarControles()
        {
            this.tbCliente.Text = string.Empty;
            this.codigoProductoSeleccionado = -1;
            this.tbProduc.Text = string.Empty;
            this.tbPrecio.Text = string.Empty;
            this.tbCant.Text = "";
            this.crearTabla();
            this.lbTotalPagar.Content = "Total Pagar: S/. 0.00";
        }

        private void mError(string mensaje)
        {
            MessageBox.Show(mensaje, "ERROR", MessageBoxButton.OK, MessageBoxImage.Information);
        }
        //Para mostrar mensaje de confirmación
        private void mOk(string mensaje)
        {
            MessageBox.Show(mensaje, "MENSAJE", MessageBoxButton.OK, MessageBoxImage.Information);
        }

        private void crearTabla()
        {
            dtDetalle = new DataTable("Detalle");
            //Agrega las columnas que tendra la tabla
            dtDetalle.Columns.Add("codproducto", typeof(Int32));
            dtDetalle.Columns.Add("producto", typeof(String));
            dtDetalle.Columns.Add("cantidad", typeof(Decimal));
            dtDetalle.Columns.Add("precio", typeof(Decimal));
            dtDetalle.Columns.Add("descuento", typeof(Decimal));
            dtDetalle.Columns.Add("subTotal", typeof(Decimal));
            //Relacionamos nuestro datagridview con nuestro datatable*/
            //dgDetalle.AutoGenerateColumns = false;
            dgDetalle.DataContext = dtDetalle;
        }

        private void Page_Loaded(object sender, RoutedEventArgs e)
        {
```

```
            this.crearTabla();
        }

        private void btBuscar_Click(object sender, RoutedEventArgs e)
        {
            SelecProducto frame = new SelecProducto();
            frame.estableceVentana(this);
            frame.ShowDialog();
        }

        private void btAgregar_Click(object sender, RoutedEventArgs e)
        {
            if (this.codigoProductoSeleccionado == -1)
            {
                this.mError("No ha seleccionado aun ningun producto");
            }
            else
            {
                //Variable que va a indicar si podemos registrar el detalle
                bool registrar = true;
                foreach (DataRow row in dtDetalle.Rows)
                {
                    if (Convert.ToInt32(row["codproducto"]) == this.codigoProductoSeleccionado)
                    {
                        registrar = false;
                        this.mError("Ya se encuentra el producto en el detalle");
                    }
                }
                //Si podemos registrar el producto en el detalle
                if (registrar)
                {
                    //Calculamos el sub total del detalle sin descuento
                    decimal subTotal = Convert.ToDecimal(this.tbPrecio.Text) *
Convert.ToDecimal(this.tbCant.Text);
                    //Obtenemos el descuento
                    decimal descuento =
NegDetalleVenta.ObtenerDescuento(Convert.ToDecimal(tbCant.Text),
Convert.ToDecimal(this.tbPrecio.Text));
                    //Actualizamos el sub total con el descuento correspondiente
                    subTotal = subTotal - descuento;
                    //Aumentamos el total a pagar
                    this.totalPagar += subTotal;
                    this.lbTotalPagar.Content = "Total Pagar: S/." + totalPagar.ToString("#0.00#");
                    //Agregamos al fila a nuestro datatable
                    DataRow row = this.dtDetalle.NewRow();
                    row["codproducto"] = this.codigoProductoSeleccionado;
                    row["producto"] = this.tbProduc.Text;
                    row["cantidad"] = this.tbCant.Text;
                    row["precio"] = this.tbPrecio.Text;
                    row["descuento"] = descuento;
                    row["subTotal"] = subTotal;
                    this.dtDetalle.Rows.Add(row);
                }
            }
        }

        private void btQuitar_Click(object sender, RoutedEventArgs e)
        {
            try
            {
                //Indice dila actualmente seleccionado y que vamos a eliminar
                int indiceFila = this.dgDetalle.SelectedIndex;
                //Fila que vamos a eliminar
                DataRow row = this.dtDetalle.Rows[indiceFila];
                //Disminuimos el total a pagar
                /*this.totalPagar = this.totalPagar -
Convert.ToDecimal(row["subTotal"].ToString());
                //this.lblTotalPagar.Text = "Total Pagar: S/." + totalPagar.ToString("#0.00#");
                //Removemos la fila
                this.dtDetalle.Rows.Remove(row);*/
                label.Content = indiceFila.ToString();
            }
            catch (Exception ex)
```

```
            {
                mError("No hay fila para remover " + ex.Message);
            }
        }

        private void tbGuardar_Click(object sender, RoutedEventArgs e)
        {
            if (this.dtDetalle.Rows.Count > 0)
            {
                string rpta = NegVenta.Insertar(this.tbCliente.Text, this.dtDetalle);
                if (rpta.Equals("OK"))
                {
                    mOk("Se inserto de manera correcta la venta");
                    this.limpiarControles();
                }
                else
                {
                    mError(rpta);
                }
            }
            else
            {
                mError("No agregado ningun detalle");
            }
        }

    }
```

Modificar la ventana principal **MainWindow**




Agregar el botón correspondiente a Ventas

```
<Button x:Name="btVentas" Content="Reg. Ventas" Width="100" Click="btVentas_Click"
FontWeight="Bold" Height="25" Margin="10,0,0,0"/>
```

Programar el boton **btVentas**

```
private void btVentas_Click(object sender, RoutedEventArgs e)
{
    Main.Content = new VentaDetalles();
}
```