



INSTITUTO TECNOLÓGICO SUPERIOR IBARRA

Nombre: Samia De La Cruz

Curso: Cuarto Desarrollo de Software

Ing. Alexandra Juma

Gestión de Bases de Datos

Tema: Ensayo Framework ADO .net

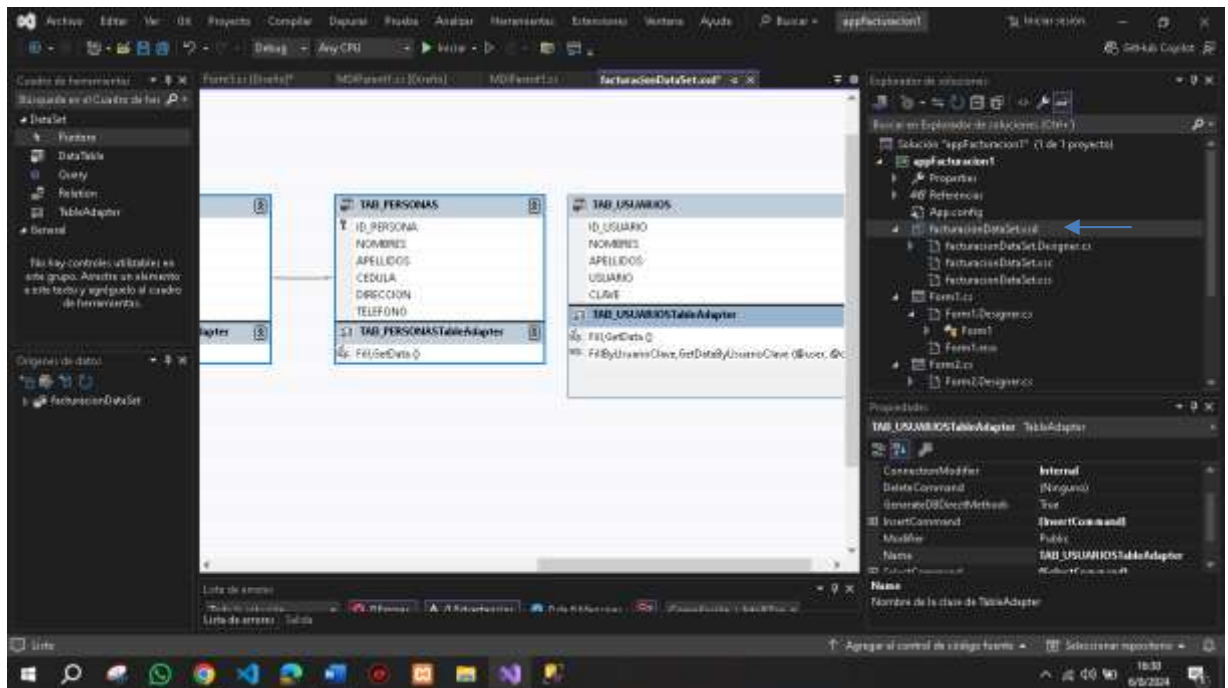
Realizar un ensayo de los Objetos principales de ADO .net con sus funcionalidades y una captura de pantalla de un ejemplo de cada objeto que se encuentre en el proyecto de Facturación que se esta realizando en clases.

Introducción

ADO.NET (ActiveX Data Objects para .NET) es una tecnología de Microsoft diseñada para facilitar el trabajo de los programadores en entornos .NET. Como menciona el autor ADO es una interfaz de programación que “incluye un conjunto de clases que proporcionan servicio de acceso a bases de datos” (Ceballos, 2015).

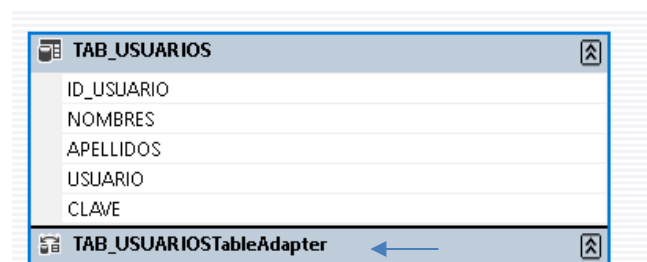
Desarrollo

En el proyecto de facturación existe un archivo facturacionDataSet.xsd: este archivo permite visualizar el diseño y definir la estructura de los datos

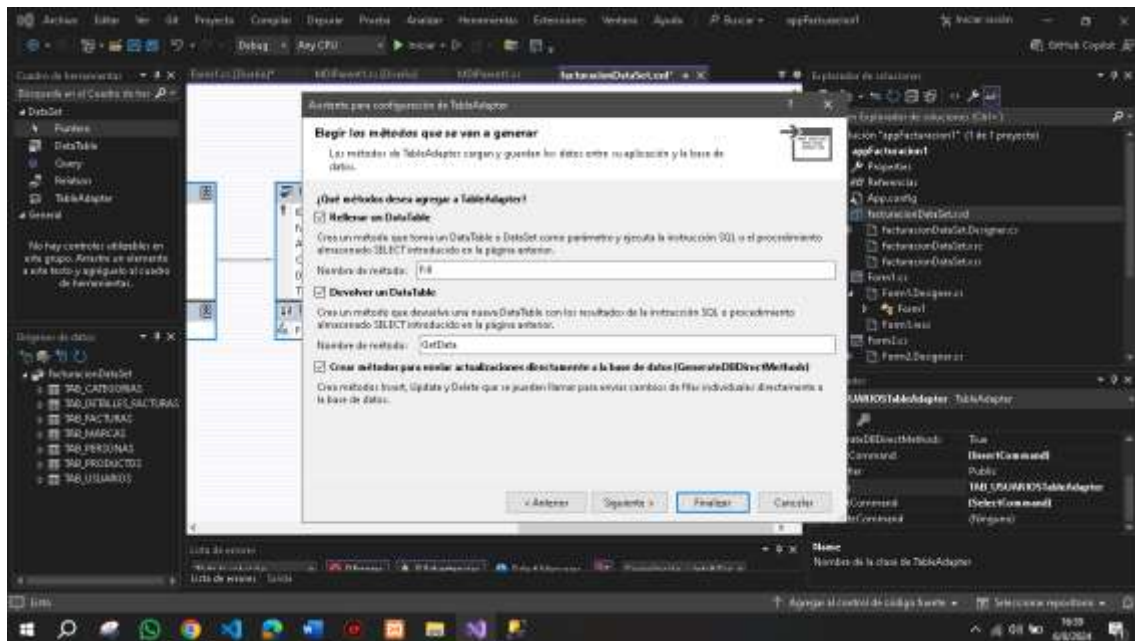


En mi tabla tab_usuarios existen 3 objetos importantes

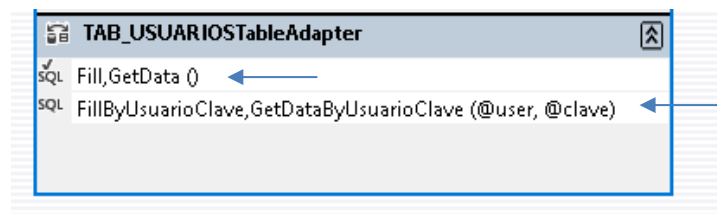
los objetos DataTable se utilizan para representar las tablas de un DataSet. Un objeto DataTable representa una tabla de datos relacionales de la memoria; los datos son locales de la aplicación basada en .NET en la que residen, pero se pueden llenar desde un origen de datos como Microsoft SQL Server mediante un DataAdapter.



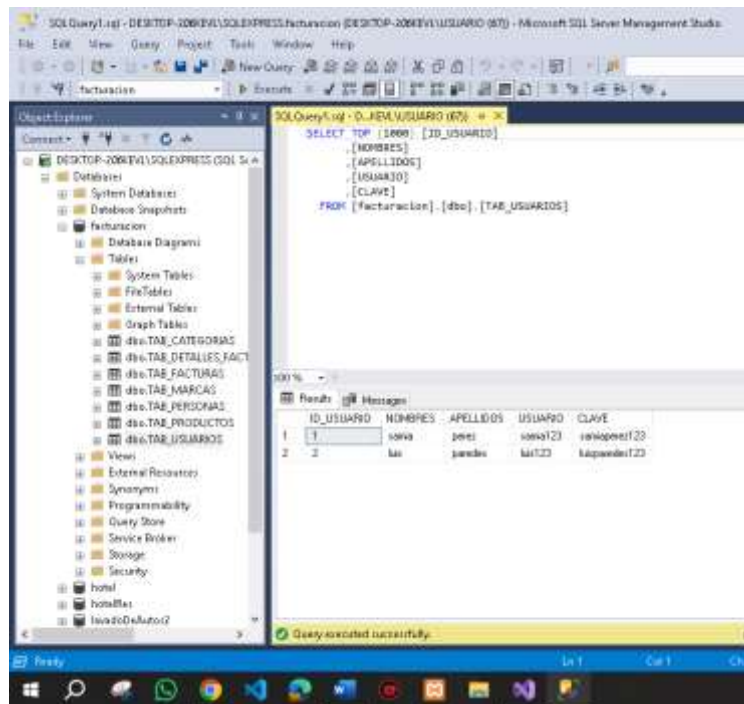
La clase TAB_USUARIOSTableAdapter es un componente del DataSet está asociado a mi tabla tab_usuarios, pero los métodos de TableAdapter nos permitirá cargar y guardar los datos entre nuestra aplicación y la base de datos.



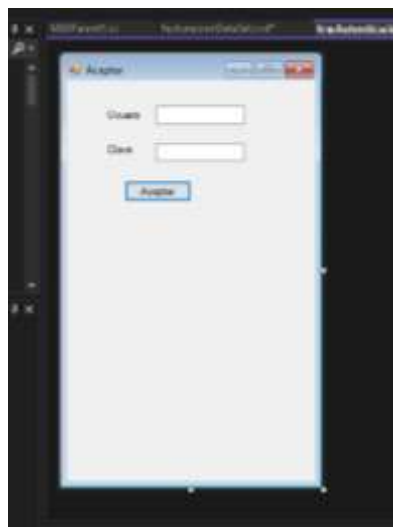
En esta parte se crea un método Fill que toma un DataTable o un DataTable como parámetro y ejecuta una consulta SQL (SELECT). Este método llena el DataTable con los datos obtenidos de la consulta. Por otro lado, el método GetData devuelve una nueva DataTable con los resultados de la consulta SQL.



En mi tabla tab_usuarios existe el método FillByUsuarioClave, GetDataByUsuarioClave(@user, @clave) es una consulta que imprime los registros de mi tabla TAB_USUARIOS y evalúa de acuerdo a mis dos parámetros @user y @clave. Esto permitirá imprimir los datos según los usuarios y las claves que tenga en mi base de datos



Para verificación tanto de mi usuario y su clave se utilizó el Formulario (Windows Forms) para el diseño y visualización de mi formulario con el nombre de frmAutenticacion



Con el cuadro de herramientas, cuadros comunes se utilizó dos Label uno para el usuario y el otro para la clave

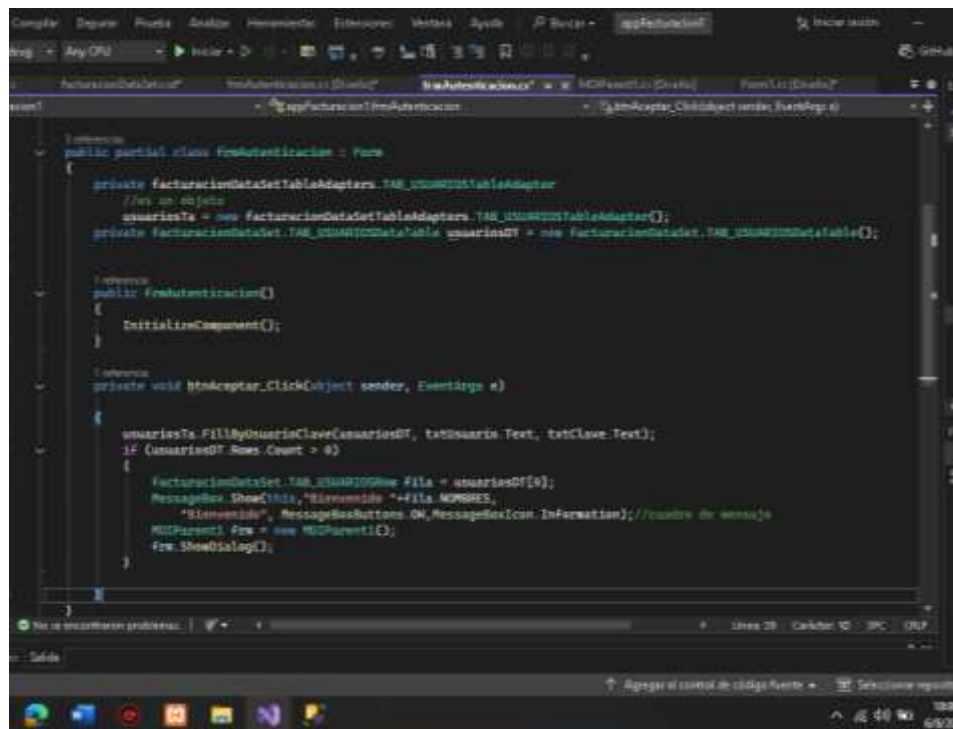
textBox dos cajas de texto donde el name se reemplazó (txtUsuario) para el usuario y (txtClave)

para la clave esto nos permitirá visualizar el texto que se ingreso en nuestros dos campos

en la clave con la propiedad PasswordChar se utiliza el elemento * para que al momento de

ingresar las credenciales solo se visualice este elemento *

Button y un botón Aceptar nos imprima el siguiente código



En el botón aceptar tenemos

```
usuariosTa = new facturacionDataSetTableAdapters.TAB_USUARIOSTableAdapter();
```

Creamos un objeto usuariosTa de tipo TAB_USUARIOSTableAdapter y se usará para relacionarse con mi tabla TAB_USUARIOS

```
private facturacionDataSet.TAB_USUARIOSDataTable usuariosDT = new facturacionDataSet.TAB_USUARIOSDataTable();
```

Después creamos un objeto usuariosDT de tipo TAB_USUARIOSDataTable. En aquí se almacenará los datos recuperados de la base de datos

```
1 referencia
private void btnAceptar_Click(object sender, EventArgs e)
{
    usuariosTa.FillByUsuarioClave(usuariosDT, txtUsuario.Text,
```

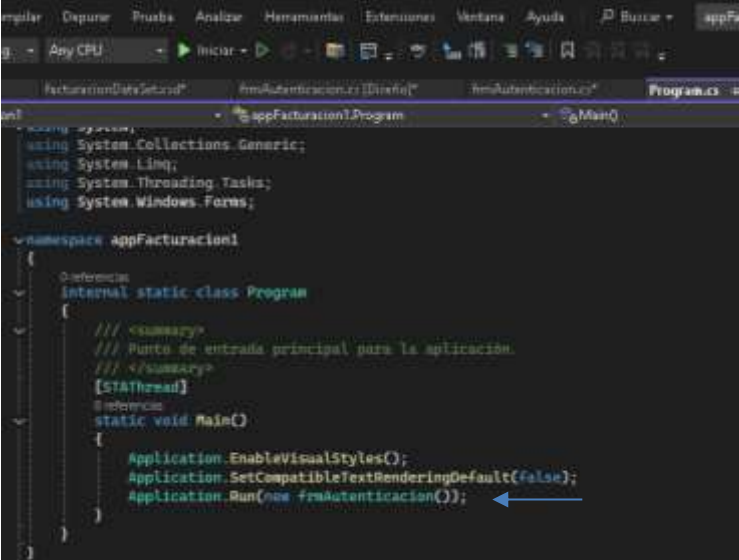
Este método maneja el evento de clic de nuestro botón Aceptar

```
usuariosTa.FillByUsuarioClave(usuariosDT, txtUsuario.Text, txtClave.Text);
```

Cuando se hace clic en el botón, se llama al método FillByUsuarioClave del TableAdapter usuariosTa, pasando usuariosDT, txtUsuario.Text y txtClave.Text como parámetros. Este método nos ejecutará la consulta SQL que definimos en FillByUsuarioClave y llena usuariosDT con los resultados

```
if (usuariosDT.Rows.Count > 0) //verifica si se encontraron registros con el usuario y clave
{
    FacturacionDataSet.TAB_USUARIOSRow fila = usuariosDT[0]; //se obtiene la primera fila del DataTable
    MessageBox.Show(this, "Bienvenido " + fila.NOMBRES,
        "Bienvenido", MessageBoxButtons.OK, MessageBoxIcon.Information); //cuadro de mensaje
    MDIParent1 frm = new MDIParent1(); //se crea una instancia del formulario MDIParent1
    frm.ShowDialog(); //se muestra como un cuadro de diálogo
}
```

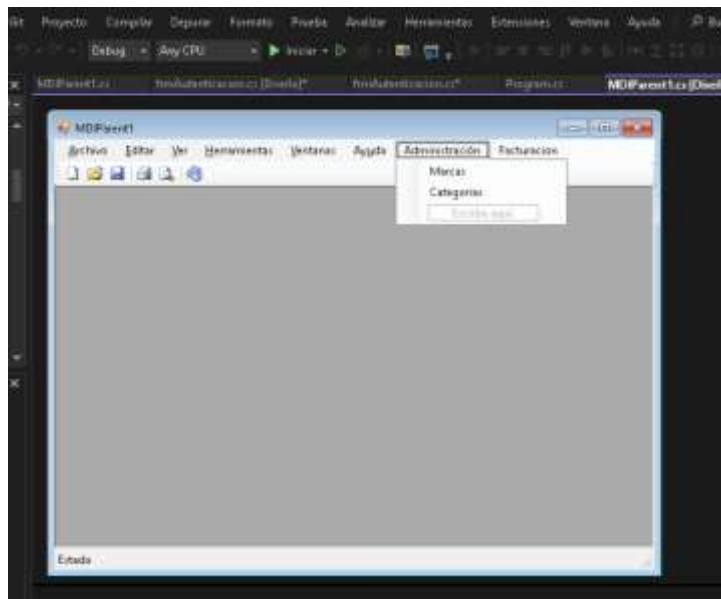
Aseguramos que al momento de ejecutar en nuestro archivo program.cs primero se ejecute el frmAutenticacion después una vez ingresado ya me muestre el formulario Primario MDI



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace appFacturacion1
{
    internal static class Program
    {
        /// <summary>
        /// Punto de entrada principal para la aplicación.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new frmAutenticacion());
        }
    }
}
```

Luego una vez ingresado con las credenciales se mostrará el formulario MDIParent1()



En el código para el manejo de los eventos al momento de dar clic

```

frmAutenticacion.cs [Diseño]*   frmAutenticacion.cs*   Program.cs   MDIParent1.cs [Diseño]*   facturacionDataSet.xsd*
+ spofacturacion1.MDIParent1   - windowMenu:

    childForm.Close();
}

1 referencia
private void marcasToolStripMenuItem_Click(object sender, EventArgs e)
//evento del clic para marcas
{
    Form childForm = new Form1(); //Crea una nueva instancia del formulario Form1
    childForm.MdiParent = this; //Establece el formulario principal MDI
    childForm.Show(); //Muestra el formulario
}

1 referencia
private void categoriasToolStripMenuItem_Click(object sender, EventArgs e)
//evento de clic para categorías
{
    Form childForm = new Form2(); //Crea una nueva instancia del formulario Form2
    childForm.MdiParent = this;
    childForm.Show(); //Muestra el formulario
}

1 referencia
private void productosToolStripMenuItem_Click(object sender, EventArgs e)
//evento de clic para productos
{
    Form childForm = new frmProductos2(); //Crea una nueva instancia del formulario frmProductos2
    childForm.MdiParent = this;
    childForm.Show(); //Muestra el formulario
}
}

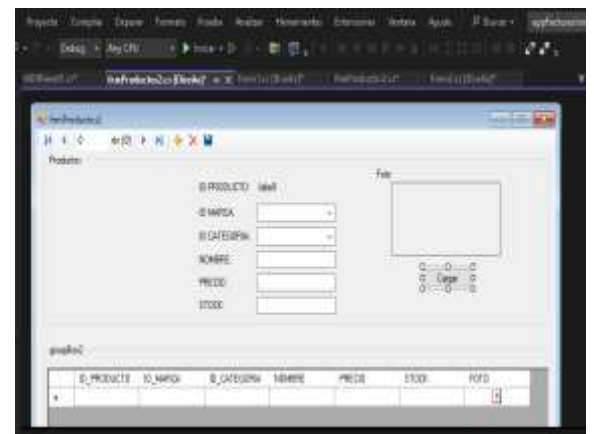
```

Para el diseño de los formularios form1, form2 y frmProductos2 y se utilizó DataSet y TableAdapters

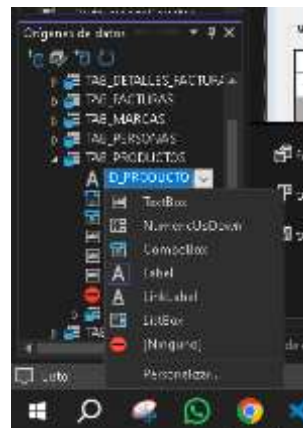
form1

form2

frmProductos2

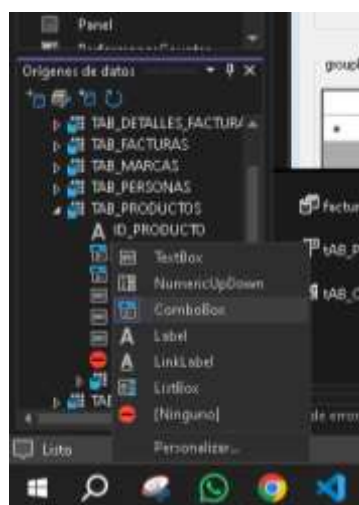


Para la tabla de mi tabla tab_productos id_producto un label

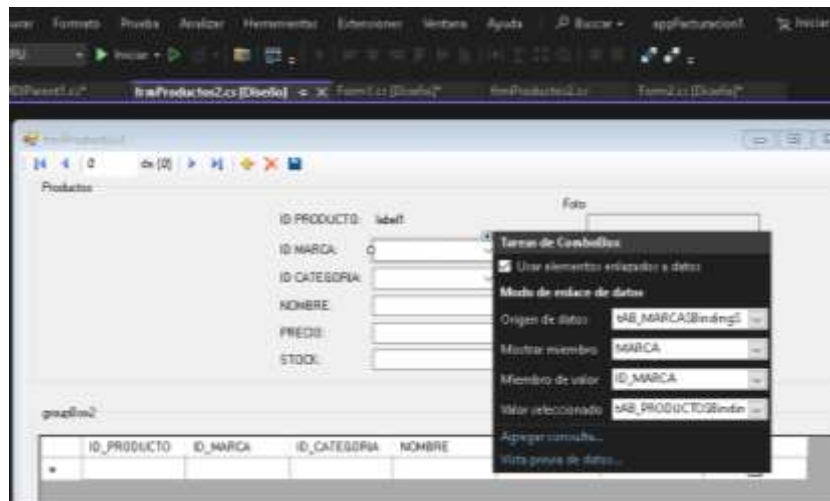


Como el id_marca y id_categoria son claves foráneas en esta tabla tab_productos seleccionamos

combo box



Después ya arrastramos a nuestro campo correspondiente



En frmProductos2 adicional se usó dos groupBox para formar grupos de botones de opción en este caso (Productos, Detalles) y un OpenFileDialog y lo nombramos como archivo

```

frmProductos2.cs [Diseño]  Form1.cs [Diseño]  frmProductos2.cs  Form2.cs [Diseño]  MDIParent1.cs [Diseño]  Program.cs
+ appFacturacion1.frmProductos2
+ TablaDataAdapter

//referencia
private void frmProductos2_Load(object sender, EventArgs e)
{
    // TODO: esta línea de código carga datos en la tabla 'FacturacionDataSet.TAB_CATEGORIAS'
    // Puede moverla o quitarla según sea necesario.
    this.TAB_CATEGORIASTableAdapter.Fill(this.facturacionDataSet.TAB_CATEGORIAS);
    // TODO: esta línea de código carga datos en la tabla 'FacturacionDataSet.TAB_MARCAS'
    // Puede moverla o quitarla según sea necesario.
    this.TAB_MARCASTableAdapter.Fill(this.facturacionDataSet.TAB_MARCAS);
    // TODO: esta línea de código carga datos en la tabla 'FacturacionDataSet.TAB_PRODUCTOS'
    // Puede moverla o quitarla según sea necesario.
    this.TAB_PRODUCTOSTableAdapter.Fill(this.facturacionDataSet.TAB_PRODUCTOS);
}

//referencia
private void btnCargar_Click(object sender, EventArgs e)
//controlador de eventos para el clic
{
    archivo.ShowDialog(); //abre el cuadro de diálogo para seleccionar una imagen
    if (archivo.FileName != null) //verifica si se ha seleccionado una imagen
    {
        pictureBox1.Image = Image.FromFile(archivo.FileName); //FromFile crea un objeto Image a partir del archivo que se selecciona
    }
}

```

Para poder cargar la imagen se implementó este código en el botón guardar

Conclusión

En mi proyecto de facturación, el uso de ADO.NET y sus componentes, como DataSet, TableAdapter y DataTable, me permitió gestionar datos y facilitar la autenticación de mis usuarios al hacer las consultas gracias a los métodos Fill y GetData. Además, estos componentes han mejorado la interacción y funcionalidad de la aplicación al permitir la carga y visualización dinámica de datos directamente desde mi base de datos facturación. Esto resultó que mis formularios sean más rápidos y fáciles de manejar tanto para mí como programador como para mis usuarios si llegan a utilizar este framework.

Bibliografía

Ceballos Sierra, F. J. (2015). Enciclopedia de Microsoft Visual Basic: interfaces gráficas y aplicaciones para Internet con Windows Forms y ASP.NET: (3 ed.). Madrid, Spain: RA-MA Editorial. Recuperado de <https://elibro.net/es/ereader/itsi/62511?page=37>.