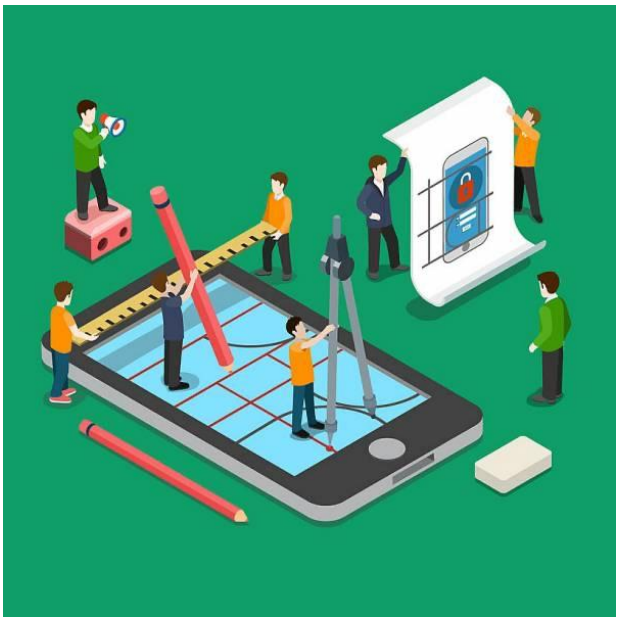
CIP de FP BATOI

DESENROTLLAMENT D'INTERFÍCIES 2N DAM

UD3: Generació d'interfícies gràfiques d'usuari.



Visual Studio C#

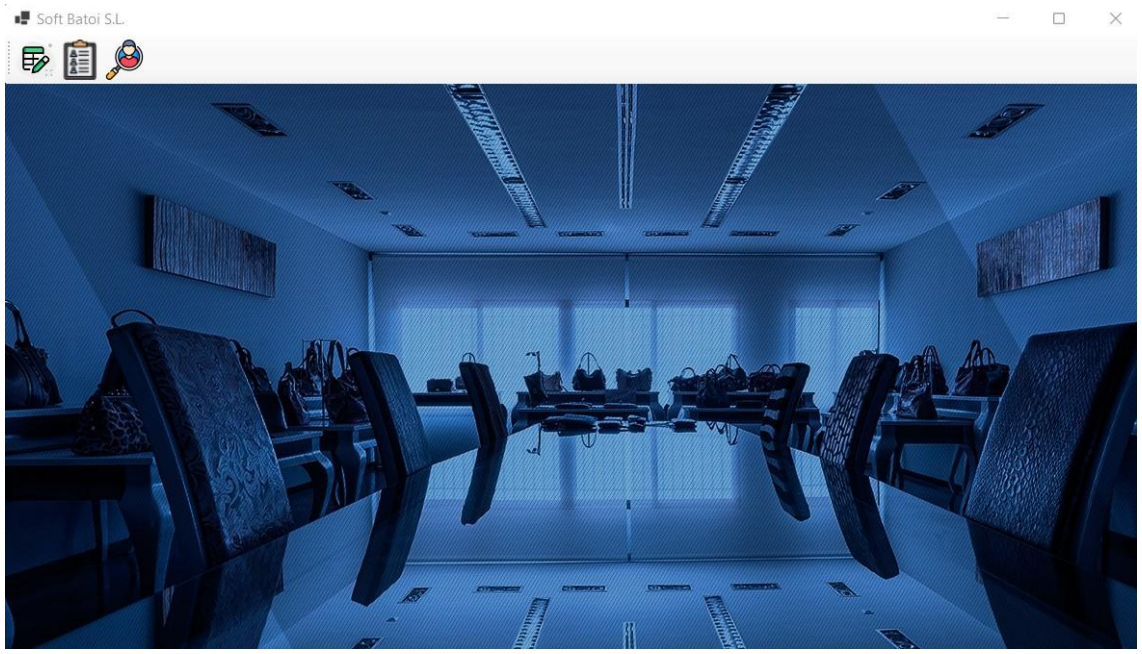
Professor: Iván Martos

Instruccions inicials
Les pràctiques es lliuraran seguint la següent pauta: El nom de l'activitat o pràctica a enviar serà el nombre de la pràctica (per exemple: A34, seguit de guió baix i del vostre nom i cognoms). Per exemple: ***A34_AntonioGómez***. L'activitat s'ha de lliurar en la data establerta i no s'acceptaran els lliuraments fora del termini establert. Les pràctiques han de ser originals i no s'admeten còpies de companys/es, Internet.. Per a les pràctiques farem ús del programari indicat per professorat i especificat a les pràctiques.

Objectius de l'activitat
1. Introduir els diferents components de Visual Studio. 2. Conèixer aspectes bàsics del llenguatge de programació C# (orientació objectes: POO...). 3. Crear aplicacions bàsiques amb Windows Forms i el llenguatge C#. 4. Entendre la gestió d'esdeveniments del llenguatge C#. **5. Aprendre a connectar amb bases de dades MySQL i MariaDB.** **6. Aprendre a connectar amb aplicacions API.**

Temporalització
L'activitat està prevista en una estimació de tres sessions lectives de 55 minuts cada sessió.

1. Anem a crear un nou projecte **responsive** en el qual disposarà d'un formulari principal amb el següent aspecte:



El formulari principal tindrà una imatge de fons similar al que heu vist abans. També disposarà d'un **toolStrip** que ens permetrà afegir una barra de ferramentes on inserirem tres botons amb les icones semblants a les que heu vist.

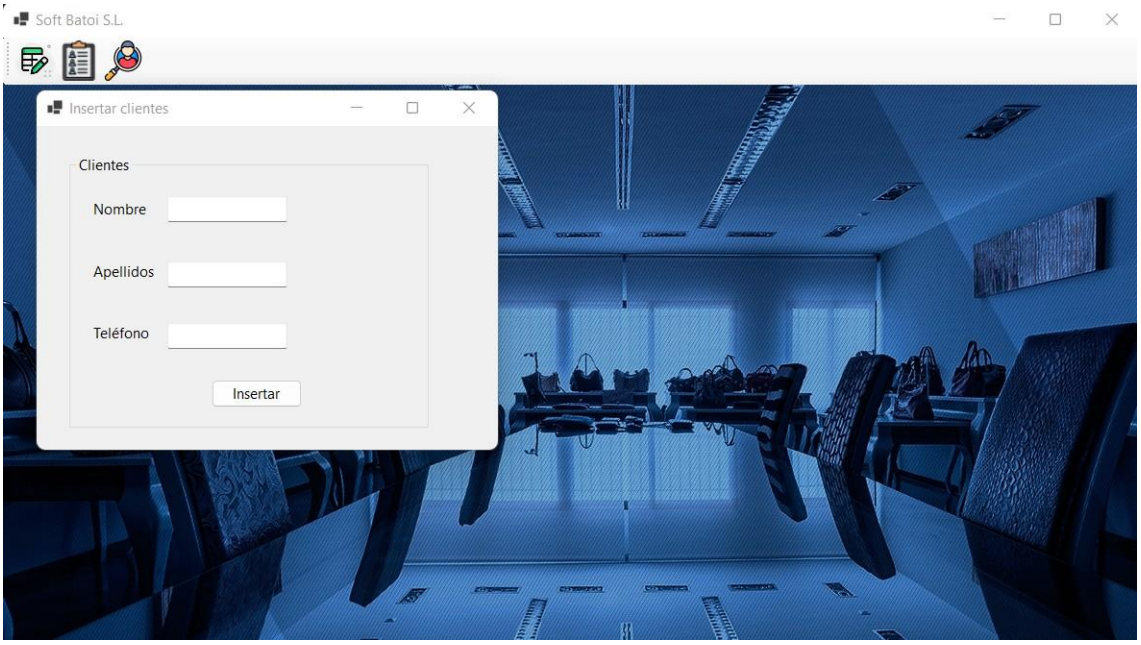
La barra de ferramentes disposarà de tres botons amb icones per a:

- Afegir un nou client.
- Buscar un client pel seu identificador o codi de client.
- Llistar els clients actuals.

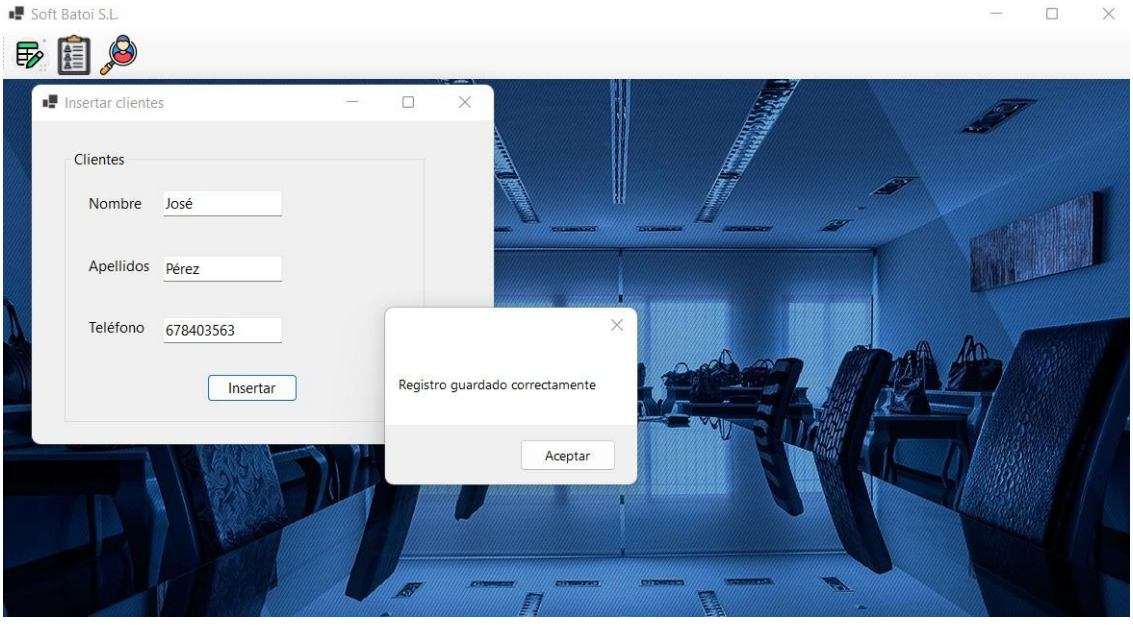
Aquesta pràctica farà ús d'una **base de dades MySQL o MariaDB** que s'anomenarà empresa i disposarà d'una taula de clients amb quatre camps bàsics (idClient, nom, cognoms i telèfon). L'identificador del client serà un enter i serà autoincremental.

Per a realitzar aquesta pràctica, recomanem fer ús de la plataforma XAMPP (integra PHP, MySQL o MariaDB, servidor web Apache...). La pàgina oficial de XAMPP és: https://www.apachefriends.org/es/index.html

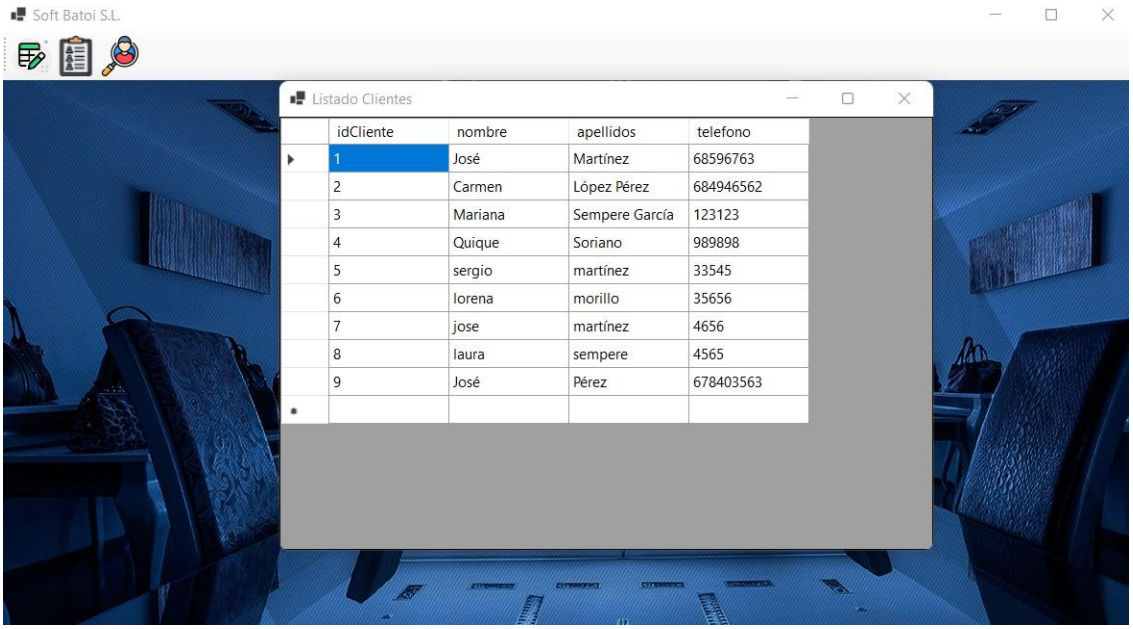
A continuació mostrarem el contingut del formulari per a inserir un nou client (recordeu que l'identificador és auto incremental i per això no inserirem el codi del client).



Una vegada emplenem les dades al formulari, aquest registre de nou client s'inserirà en la base de dades client i en la seua corresponent taula empresa. Per això, mostrarem un missatge informatiu indicant que s'ha emmagatzemat el registre correctament.

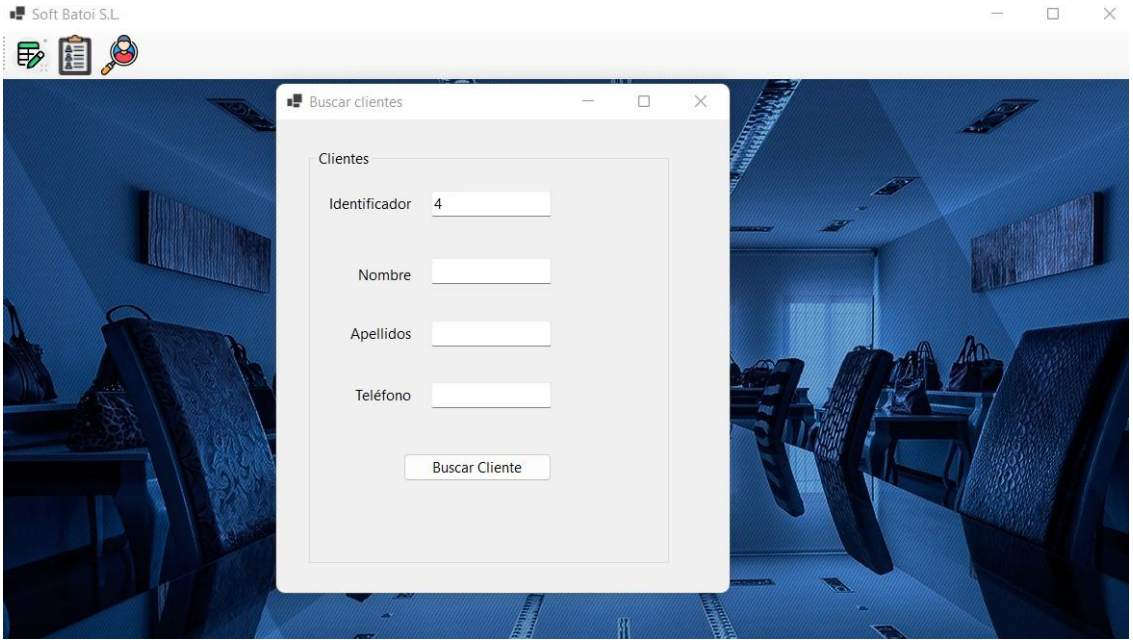


Respecte al formulari de llistar clients, inserirem un **datagridview** que agafarà les dades de la taula client de la base de dades empresa.

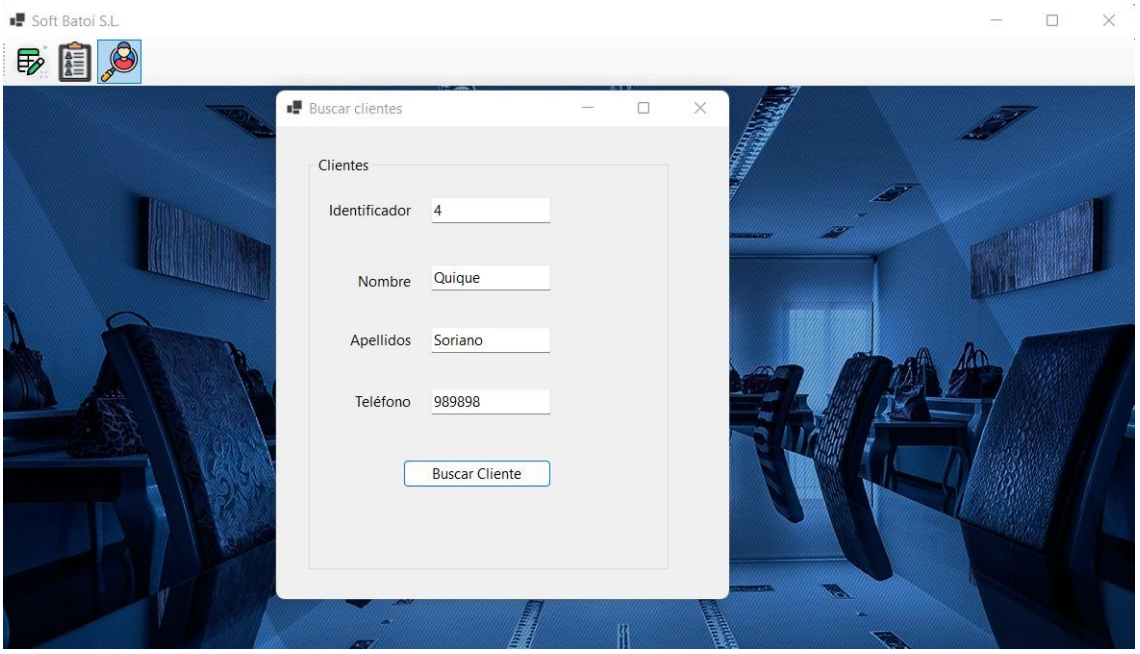


Finalment, ens queda l'opció de cerca d'un determinat client atenent al seu identificador. Caldrà fer una còpia del formulari d'Inserir un nou client, ja que és molt semblant. Com podeu veure, sols afegirem el camp de l'identificador. Si escrivim un identificador, en clicar s'emplenarà la resta d'informació del formulari (nom, cognoms i telèfon del client en qüestió, sempre que existisca un client amb eixe identificador).

En l'exemple, hem inserit un identificador de client i ens queda clicar al botó de buscar client per a recuperar la resta de camps del client amb l'identificador 4.



Ací tenim el resultat, una vegada hem clicar a buscar client. Com podeu observar ha recuperat els camps corresponents al client en qüestió (nom, cognoms i telèfon).



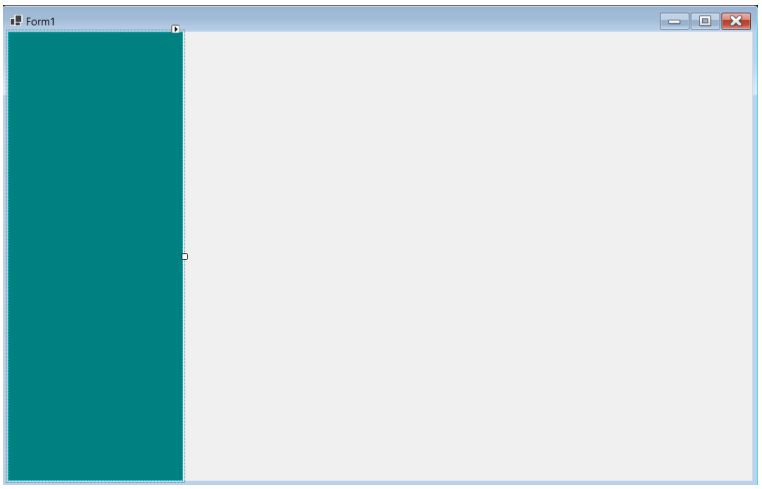
Si fiquem un id que no existeix, ha de eixir un missatge informatiu indicant que el id no existeix.

2. Creació de menús laterals i personalització de la interfície.

Crearem un projecte **responsive** amb **Visual Studio C#** i **Windows Forms**. En aquesta pràctica anem a personalitzar l'aparença de la nostra aplicació incloent un **menú lateral** per així veure diferents alternatives als menús clàssics que ja hem utilitzat a pràctiques anteriors.

Crearem un menú lateral fent ús del component **Panel**. El panel estarà situat al **lateral esquerra** del formulari principal i caldrà canviar el nom al component per a facilitar la seua identificació (per exemple **panelPrincipal**).

Propietat	Valors
Name	panelPrincipal
Dock	Left
Size	250px (Amplària)
Backcolor	Teal

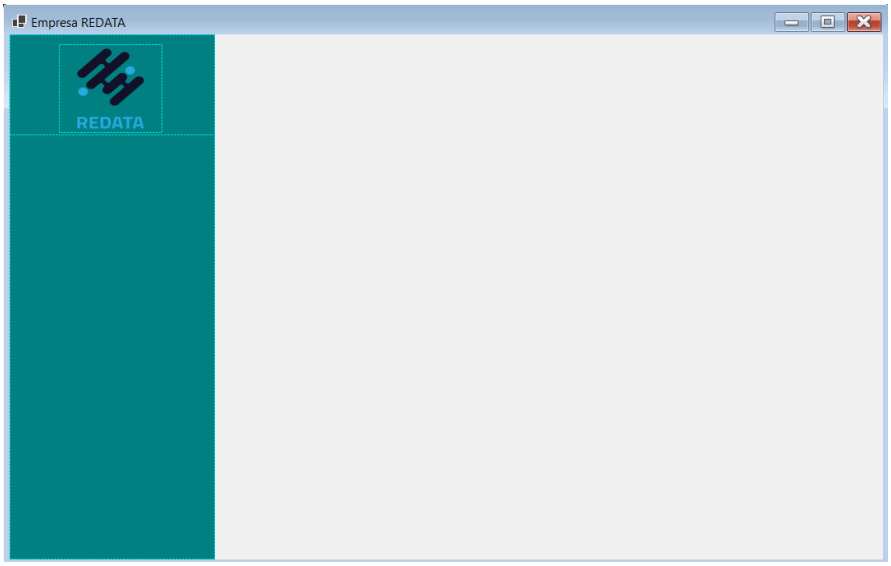


Ara crearem un nou panel dintre del panel lateral principal. Aquest nou panel ens servirà per afegir el **logotip de l'empresa** en qüestió.

Les propietats del nou panel que acabem de crear dintre del principal son:

Propietat	Valors
Name	panelLogo
Dock	Top
Size	250px; 123px

Respecte al panel anterior, li afegirem un component **PictureBox** i li afegirem el logotip de l'empresa.



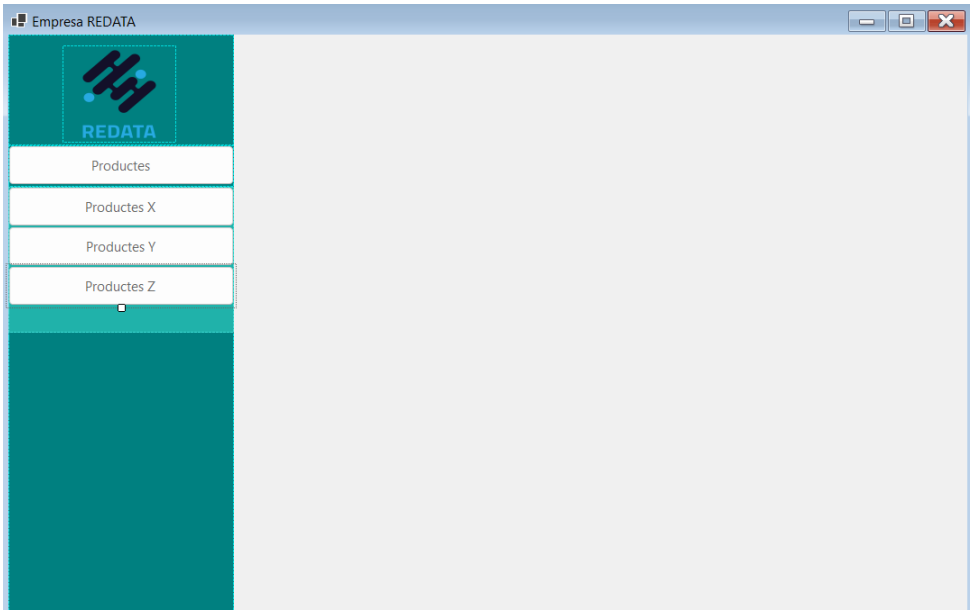
Crearem un nou panel que estarà situat baix del panel del logotip. En aquest panel (**panelProductes),** que tindrà la amplària per a albergar a soles un botó**,** anem a afegir un **botó (botoProductes)** que ens permetrà obrir un submenú en aquest cas de Productes.

Propietat	Valors
Name	botoProductes
Dock	Top
Text	Productes
Size	44 px (altura del botó)

Ara crearem un nou panel que serà per a mostrar els botons corresponent al submenú de productes. El nou panel del submenú s'anomenarà panelSubProductes i canviarem el color de fons a color LightSeaGreen. En aquest panel afegirem tres botons que formaran part del submenú de Productes. La amplària d'aquest panel serà la suficient per a albergar tres botons.

Propietat	Valors
Name	panelSubProductes
Dock	Top
BackColor	LightSeaGreen

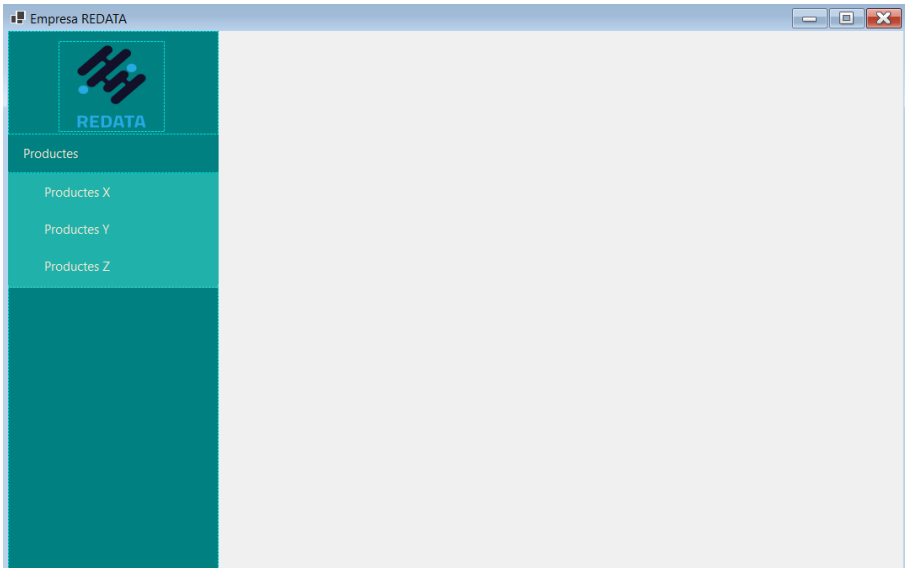
Ara crearem dintre del panel per al submenú tres botons diferents amb la propietat **Dock a Top** i amb una altura de **44px**. En aquesta pràctica com estem treballant més l'aspecte visual, els productes seran genèrics i posarem X,Y,Z en els botons, però posteriorment ho podem canviar i personalitzar, així com tots els colors que son simplement exemples per a modificar l'aspecte i l'aparença de les nostres aplicacions.



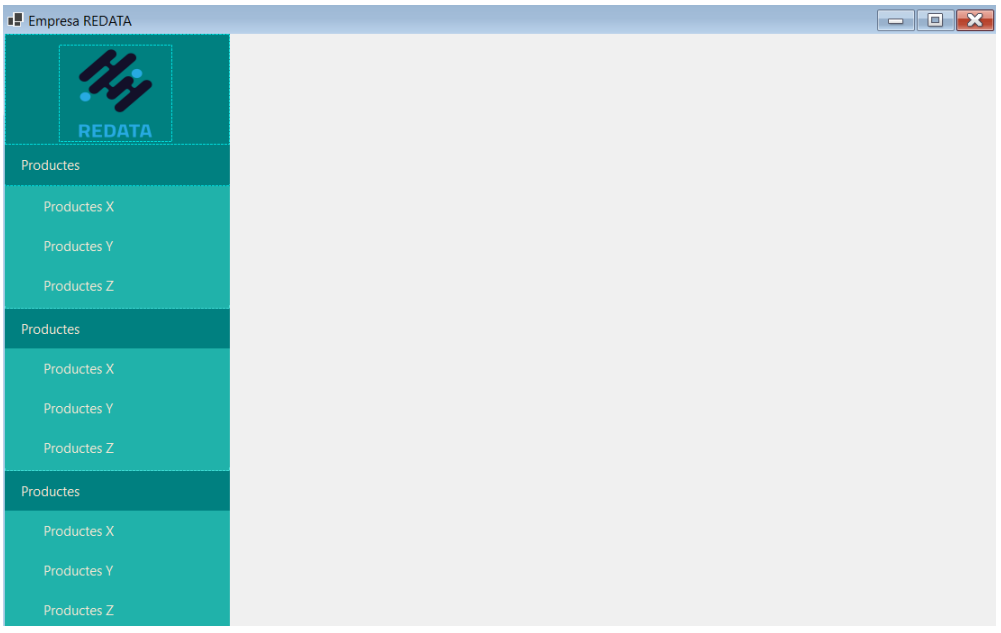
Respecte a **tots els botons** creats fins ara, canviarem les següents propietats:

Propietat	Valors
FlatStyle	Flat
FlatAppareance (BorderSize)	0
ForeColor	AntiqueWhite
BackColor	LightSeaGreen
TextAlign	MiddleLeft
Padding (Left)	• 10px (botó principal de productes) • 35 px (resta de botons de Productes: X,Y,Z)

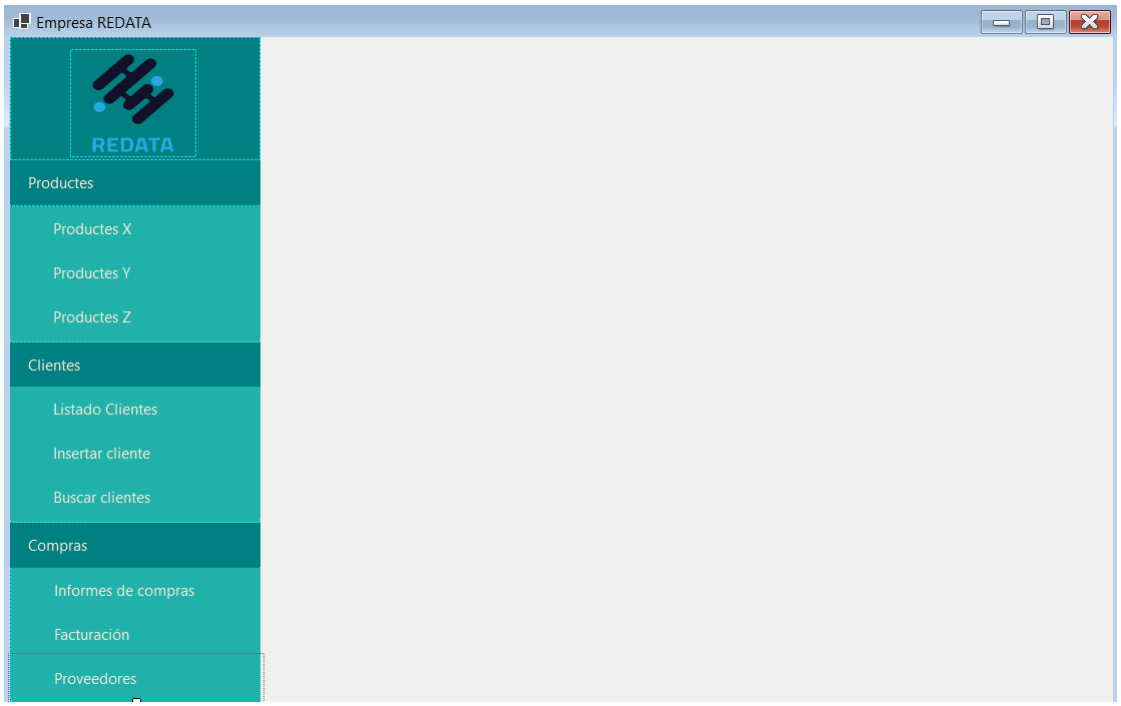
A continuació veiem l'aspecte visual després d'aplicar les propietats sobre els botons.



Ara el que farem és replicar la mateixa estructura per a tindre tres botons amb els seus corresponents submenús.



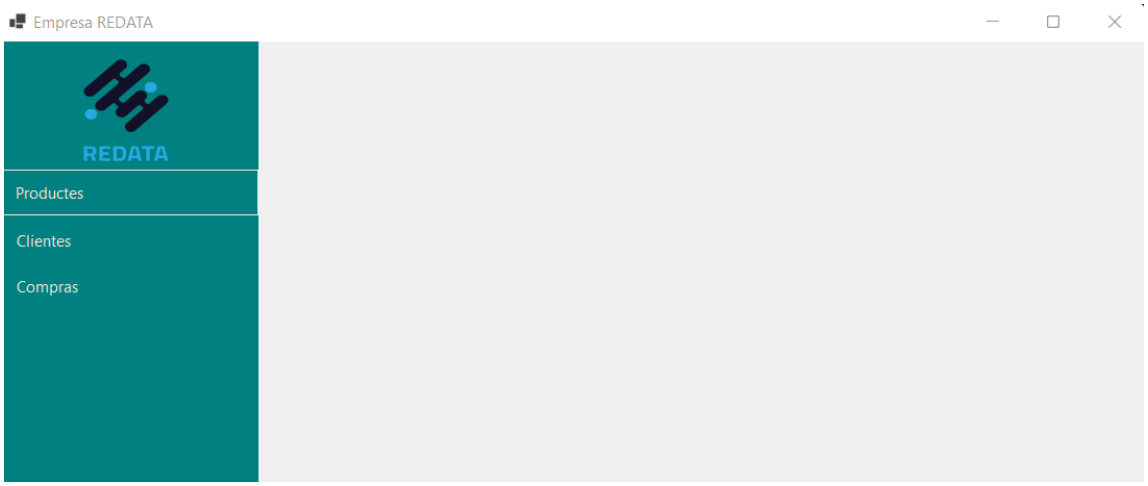
Ara ho adaptarem per canviar els botons i els submenús, recordeu canviar tant els textos com el **nom del components i dels panels**, ja que després ens facilitarà la localització del codi, la solució d'errors i la programació identificant correctament cada element. També posarem el **AutoScroll** del panelPrincipal a **vertader** per així en cas de canviar la grandària del formulari, el menú és puga veure desplaçant l'escroll.



En aquest punt, encara que podem canviar i personalitzar més la nostra aplicació, per exemple canviar els botons de productes... (ho podem fer posteriorment), anem a començar a programar el codi per a modificar el comportament de l'aplicació.

Crearem un **mètode** per a **ocultar tots els panels dels submenús** per a què en el moment d'iniciar l'aplicació, no es visualitzen els submenús, únicament els botons dels menús principal, és a dir, Productes, Clients i Compres.

Aquesta és l'aparença que volem en iniciar l'aplicació (totes les opcions del submenú estan ocultes).



Per això, podem programar un mètode o directament o podem posar al constructor on ocultem tots els panels dels submenús. Recordeu que és molt important que cada panel tinga un nom identificatiu per a facilitar la seua gestió:

```
1 referencia
public Form1()
{
    InitializeComponent();
    inici();
}

1 referencia
private void inici()
{
    panelSubProductes.Visible = false;
    panelSubClients.Visible = false;
    panelSubCompras.Visible = false;
}
```

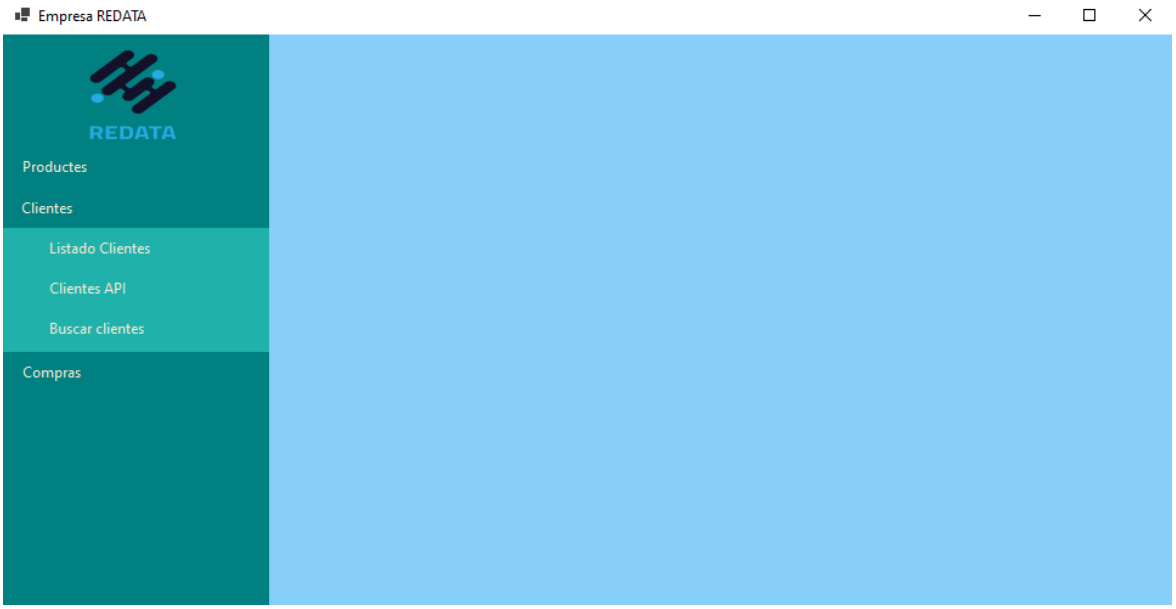
També seria convenient disposar d'un mètode que comprove si hi ha algun panel que estiga visible i que l'oculte.

Després crearem un mètode que mostre el submenú que corresponga. Per això, podem passar-li com a paràmetre el panel i comprovar si és visible per mostrar-lo o ocultar-lo si ja estava mostrat.

A continuació, mostrem un fragment de codi de com podria quedar. El mètode d'ocultar submenús l'hem de programar nosaltres abans.

```
private void showSubmenu(Panel subMenu)
{
    if (!subMenu.Visible)
    {
        ocultarSubmenus();
        subMenu.Visible = true;
    }
    else
    {
        subMenu.Visible = false;
    }
}
```

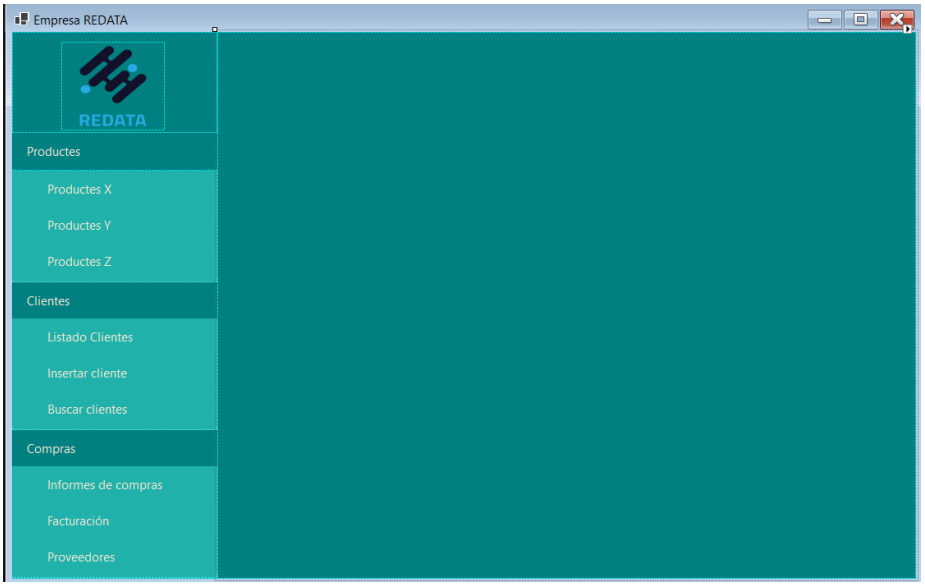
Ara a través dels botons principals, ja podem invocar als mètodes per a mostrar els corresponents submenús. Ací mostrem un exemple on al clicar en el botó principal de Clients, ha mostrat el submenú de clients. En cas de tornar a cridar, s'ocultarà el submenú en qüestió.



Ací veiem el codi que podem fer ús per a que al clicar en els botons principals (Productes, Clients i Compres) li passem el panel que volem mostrar o ocultar.

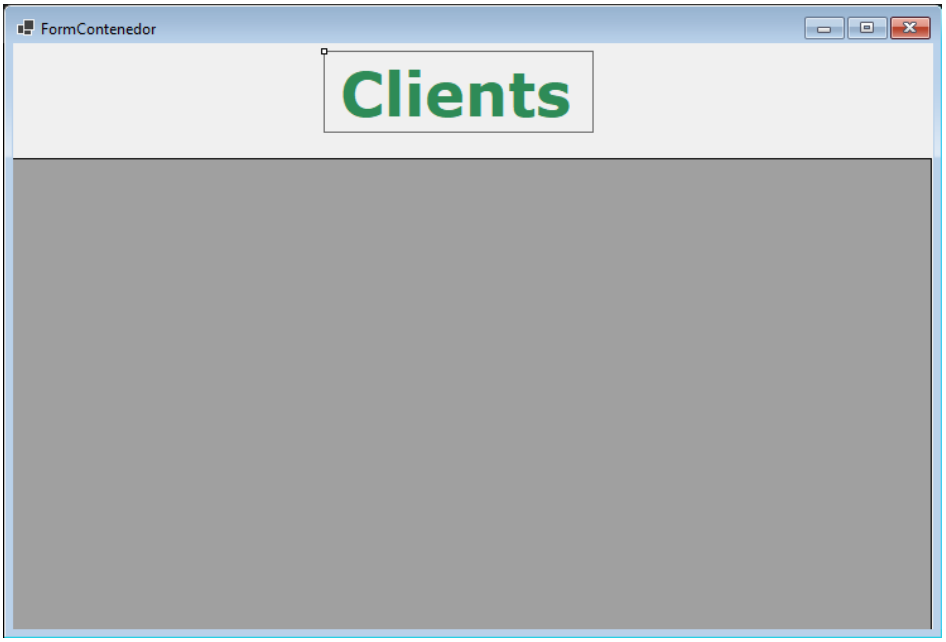
```
private void botoProductes_Click(object sender, EventArgs e)
{
    showSubmenu(panelSubProductes);
}

1 referencia
private void botoClients_Click(object sender, EventArgs e)
{
    showSubmenu(panelSubClients);
}

1 referencia
private void botoCompras_Click(object sender, EventArgs e)
{
    showSubmenu(panelSubCompras);
}
```

Ara creem un nou panel **(panelContenidor**) per a la part dreta que serà la contenidora on podrem mostrar la informació. Li aplicarem el color de fons "Teal" per a què tinga el mateix color que el menú (encara que seria millor un color més claret). Li donem Dock Fill per a que ocupe la resta d'espai. Podem personalitzar els colors al nostre gust, ja que açò simplement és un exemple de com podem crear els submenús.



Una vegada que ja tenim el contenidor (**panelContenidor**) de la dreta on apareixerà la informació una vegada cliquem en qualsevol botó, **crearem un nou formulari** amb la mateix grandària que el contenidor (panelContenidor). Aquest formulari nou serà el que s'obriga sobre el panelContenidor.

El seu aspecte pot ser el següent, a soles conté un label i un dataGridView.



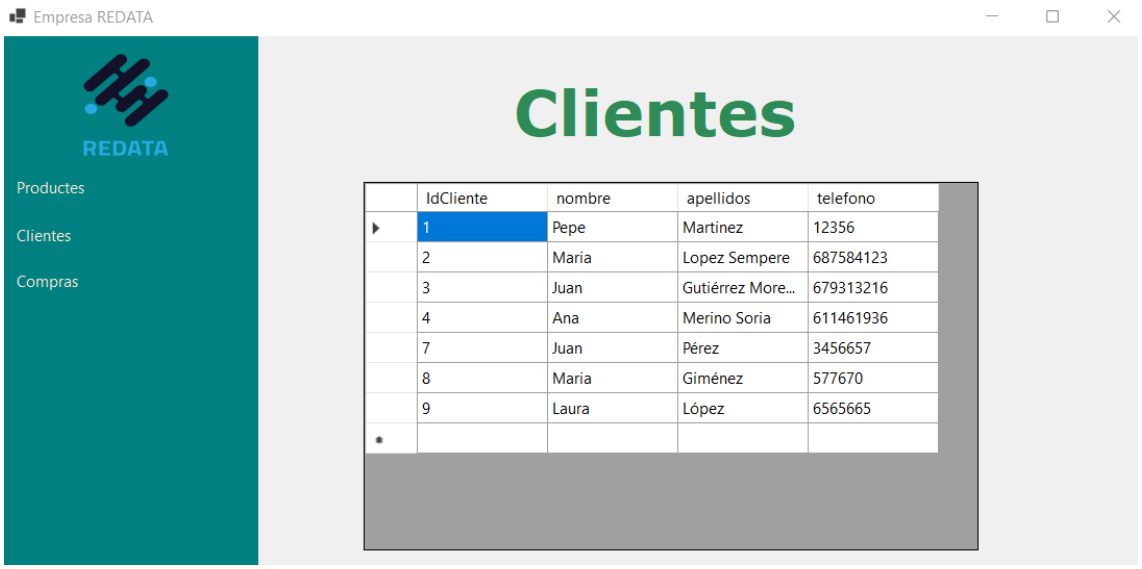
Ara programarem el següent còdec per a crear un mètode que permeta mostrar el nou formulari que volem segons el botó que hem clicat.

```
private Form formularioActivo = null;

// Método para abrir un formulario hijo dentro de un panel contenedor
1 referencia
private void abrirFormulario(Form hijo)
{
    // Si ya hay un formulario activo, lo cerramos antes de abrir uno nuevo
    if (formularioActivo != null)
    {
        formularioActivo.Close(); // Cierra el formulario actualmente activo
    }

    // Asignamos el nuevo formulario hijo a la variable formularioActivo
    formularioActivo = hijo;

    // Configuramos el formulario hijo para que no sea independiente
    hijo.TopLevel = false; // Indica que el formulario no es una ventana principal

    // Configuramos el formulario hijo para que ocupe todo el espacio del contenedor
    hijo.Dock = DockStyle.Fill; // Hace que el formulario se ajuste al tamaño del contenedor

    // Quitamos los bordes del formulario hijo (sin barra de titulo ni bordes redimensionables)
    hijo.FormBorderStyle = FormBorderStyle.None;

    // Configuramos el panel contenedor para que también ocupe todo el espacio disponible
    panelContenedor.Dock = DockStyle.Fill;

    // Agregamos el formulario hijo al contenedor como un control
    panelContenedor.Controls.Add(hijo);

    // Asociamos el formulario hijo al panel a través de la propiedad Tag (por si lo necesitamos después)
    panelContenedor.Tag = hijo;

    // Traemos el formulario hijo al frente para asegurarnos de que sea visible
    hijo.BringToFront();

    // Mostramos el formulario hijo dentro del contenedor
    hijo.Show();
}
```

Després des de cada botó podem cridar al mètode al que li passarem el formulari que volem mostrar.

```
private void btListarClientes_Click(object sender, EventArgs e)
{
    abrirFormulario(new FormContenedor());
    ocultarSubmenus();
}
```

El que finalment ens queda es mostrar el clients en el formulari nou una vegada cliquem sobre el botó de Listado clientes.
Ací veiem el resultat on es mostra el dataGridView per a visualitzar els clients emmagatzemats en una base dades amb el sistema gestor de base de dades MySQL.



3. Connexió a API REST amb C# i WinForms

3.1. HTTPClient

HttpClient és la classe de .NET que permet fer peticions HTTP de manera senzilla i asíncrona.
Funciona molt bé amb **async i await.**
Per utilitzar-lo:

```
using System.Net.Http;
using System.Text.Json;
```

I es recomana tindre una sola instància:

```
private readonly HttpClient _httpClient = new HttpClient();
```

3.2. Fer una petició GET

Exemple bàsic per obtindre usuaris des d'una API pública:

```
string url = "https://jsonplaceholder.typicode.com/users";

var response = await _httpClient.GetAsync(url);
response.EnsureSuccessStatusCode();

string json = await response.Content.ReadAsStringAsync();

var options = new JsonSerializerOptions
{
```

```
    PropertyNameCaseInsensitive = true
};

List<ApiUser> users = JsonSerializer.Deserialize<List<ApiUser>>(json, options);
```

Explicació:

1. **GetAsync(url)** → envia la petició GET.
2. **EnsureSuccessStatusCode()** → comprova que la resposta siga 2xx.
3. **ReadAsStringAsync()** → obté el JSON com a text (string).
4. **Deserialize** → converteix el JSON en una llista d'objectes C#.

3.3. Classe model per a deserialitzar JSON

```
public class ApiUser
{
    public int Id { get; set; }
    public string Name { get; set; }
    public string Email { get; set; }
}
```

La deserialització assigna automàticament el JSON a estes propietats.

3.4. Mostrar les dades en un DataGridView personalitzades

Podem personalitzar les dades a mostrar al DataGridView:

```
var data = users.Select(user => new
{
    user.Id,
    Nom = user.Name,
    Email = user.Email
}).ToList();

dataGridView1.DataSource = data;
```

3.5. Fer una petició POST

Per enviar dades a una API:

```
var nouClient = new
{
    name = "Client Prova",
    email = "client@example.com"
};

string json = JsonSerializer.Serialize(nouClient);

var content = new StringContent(json, Encoding.UTF8, "application/json");
```

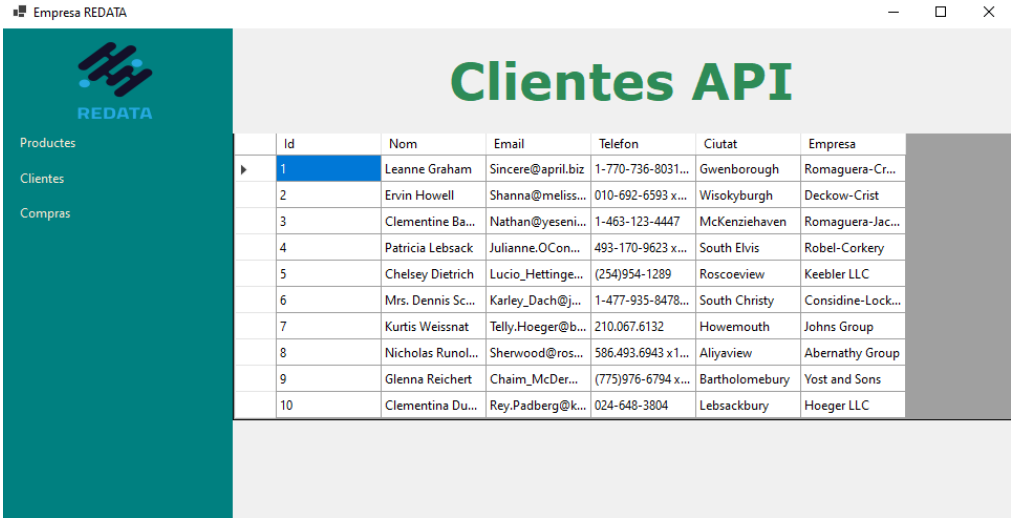
```
var response = await
_httpClient.PostAsync("https://jsonplaceholder.typicode.com/users", content);

response.EnsureSuccessStatusCode();

//Compruebo la respuesta (no es necesario)

string resposta = await response.Content.ReadAsStringAsync();
MessageBox.Show(resposta);
```

3.6. Exemple complet per a WinForms

```
private async void btnCarregarApi_Click(object sender, EventArgs e)
{
    try
    {
        string url = "https://jsonplaceholder.typicode.com/users";

        var response = await _httpClient.GetAsync(url);
        response.EnsureSuccessStatusCode(); // si no es 2xx → lanza excepción

        string json = await response.Content.ReadAsStringAsync();

        var options = new JsonSerializerOptions
        {
            PropertyNameCaseInsensitive = true
        };

        var users = JsonSerializer.Deserialize<List<ApiUser>>(json, options);

        dataGridView1.DataSource = users;

        // Opcional para autoajustar columnas:
        dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
    }
    catch (HttpRequestException httpEx)
    {
        MessageBox.Show("Error HTTP en cridar a l'API:\n" + httpEx.Message);
    }
    catch (JsonException jsonEx)
    {
        MessageBox.Show("Error al deserialitzar el JSON:\n" + jsonEx.Message);
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error inesperat:\n" + ex.Message);
    }
}
```

Aquest exemple mostrarà totes les dades i camps que conté la classe User al DataGridView, si voleu personalitzar les dades o camps a mostrar heu de fer servir el punt 3.4.

Exemple a la nostra aplicació REDATA, apartat Clients API:



Id	Nom	Email	Telefon	Ciutat	Empresa
1	Leanne Graham	Sincere@april.biz	1-770-736-8031...	Gwenborough	Romaguera-Cr...
2	Ervin Howell	Shanna@meliss...	010-692-6593 x...	Wisokyburgh	Deckow-Crist
3	Clementine Ba...	Nathan@yeseni...	1-463-123-4447	McKenziehaven	Romaguera-Jac...
4	Patricia Lebsack	Julianne.OCon...	493-170-9623 x...	South Elvis	Robel-Corkery
5	Chelsey Dietrich	Lucio_Hettinge...	(254)954-1289	Roscoeview	Keebler LLC
6	Mrs. Dennis Sc...	Karley_Dach@j...	1-477-935-8478...	South Christy	Considine-Lock...
7	Kurtis Weissnat	Telly.Hoeger@b...	210.067.6132	Howemouth	Johns Group
8	Nicholas Runol...	Sherwood@ros...	586.493.6943 x1...	Aliyaview	Abernathy Group
9	Glenna Reichert	Chaim_McDer...	(775)976-6794 x...	Bartholomebury	Yost and Sons
10	Clementina Du...	Rey.Padberg@k...	024-648-3804	Lebsackbury	Hoeger LLC

3.7. Més documentació:

Podeu ampliar informació als següents enllaços:

- https://learn.microsoft.com/en-us/dotnet/api/system.net.http.httpclient?view=net-10.0
- https://learn.microsoft.com/en-us/dotnet/fundamentals/networking/http/httpclient

ANNEX 1

Crearem la classe Connexió per a connectar amb la base de dades MySQL/MariaDB

Aquesta classe Connexió disposarà d'un mètode que permetrà connectar amb la base de dades MySQL/MariaDB i per això abans configurarem la cadena de connexió.

```
namespace Proves_BD
{
    0 referencias
    internal class conexion
    {
        0 referencias
        public static MySqlConnection conexio()
        {
            string servidor = "localhost";
            string bd = "cotxes";
            string usuario = "root";
            string pass = "";

            string cadenaConexion = "Database=" + bd + "; Data Source=" + servidor + ";"
                + "User Id= " + usuario + ";Password=" + pass;

            MySqlConnection conexionBD;

            try
            {
                conexionBD = new MySqlConnection(cadenaConexion);
                return conexionBD;


            }
            catch (MySqlException e)
            {
                Console.WriteLine("Error en la conexión " + e.Message);

            }
            return null;
        }
    }
}
```

Classe model per a deserialitzar JSON. Classe APIUser

```
public class ApiUser
{
    public int Id { get; set; }
    public string Name { get; set; }
    public string Username { get; set; }
    public string Email { get; set; }
    public ApiAddress Address { get; set; }
    public string Phone { get; set; }
    public ApiCompany Company { get; set; }
}

public class ApiAddress
{
    public string Street { get; set; }
    public string Suite { get; set; }
    public string City { get; set; }
    public string Zipcode { get; set; }
}

public class ApiCompany
{
    public string Name { get; set; }
}
```

Exercici:

Heu de fer la aplicació 1, taller de cotxes y la segona aplicació REDATA.

En la aplicació REDATA el Apartat Listado clientes ha de mostrar el cliets de la base de dades MySQL i el apartat Clients API ha de mostrar els users de la plataforma JSON Place Holder.

Puntuacions del exercici:

Apartat	Puntuació
Aspecte visual de la interfície 1, amb totes les funcionalitats i responsive.	1 punt
Inserció de clients a la base de dades i els missatges informatius.	1 punt
Es mostren els clients correctament al dataGridWiew.	1 punt
Funciona la cerca de clients i les excepcions.	1 punt
Aspecte visual de la interfície 2, amb totes les funcionalitats i responsive.	1 punt

Funcionalitat dels botons, es despleguen i es pleguen correctament.	1 punt
AutoScroll, true.	0,5 punts
Carrega del formulari contenidor correctament.	1 punt
El dataGridView del formulari contenidor mostra les dades de la base de dades MySQL.	0,5 punts
El dataGridView del formulari contenidor API Rest mostra les dades del usuaris de JSON Place Holder.	1,5 punts
Claredat del codi, documentació del codi, llegibilitat, bon funcionament, execució sense errors, fidelitat en l'aparença, correcta estructuració, modularitat...	0,5 punts