

(DISEÑO)
APLICACIÓN DE ESCRITORIO
INTERFAZ GRÁFICA COLUMNA DE FLOTACIÓN

ESTUDIANTES
Bermejo Fernández Indira Del Carmen
Padilla Maza David Antonio

Director
Jorge Luis Piñeres Mendoza

UNIVERSIDAD DEL ATLÁNTICO
PROGRAMA DE INGENIERÍA QUÍMICA
FACULTAD DE INGENIERÍA
2024

CONTENIDO

1. DISEÑO DE LA INTERFAZ	3
2. Creación de un proyecto en Visual Studio Code	4
3. Creación de una aplicación	6
4. Ejecución para el módulo automático	8
5. Ejecución herramientas	9
6. Timer	11
7. Insertar imágenes	11
8. Ejecución de la aplicación	12
9. Almacenamiento de datos	13

1. Diseño de la interfaz

1.1 Introducción

La siguiente interfaz está diseñada como una aplicación de escritorio para conocer la caída de presión que se registra en la columna de flotación ubicada en el laboratorio de carboquímica en la universidad del Atlántico. La herramienta nos permite visualizar parámetros como el Airholdup, diámetro de la burbuja, Numero de Reynolds, etc, los cuales son importantes para realizar un correcto análisis hidrodinámico.

1.2 Requisitos mínimos de Visual Studio Code 2022 versión Community

- ✚ Se admite en sistemas operativos como: Windows 7, Windows Server, Windows Server 2008 Service Pack 2, Windows Server 2008 R2 Service Pack 1, Windows Server 2012 R2, y Windows Server 2012, Windows 8.1, Windows 10 en ARM, edición LTSC de Windows 10 Enterprise.
- ✚ Para x86 o AMD64/x64, se requiere un procesador mayor o igual a 1,6 GHz.
- ✚ Se requiere 1 GB de RAM (1,5 GB si se ejecuta en una máquina virtual).
- ✚ Se requiere 1 GB de espacio disponible en el disco duro.
- ✚ Se requiere la resolución de pantalla de 1024 por 768 o superior.
- ✚ Para obtener la mejor experiencia, use la actualización más reciente de las herramientas de diagnóstico para su versión de Visual Studio.

1.3 Herramientas utilizadas durante el proyecto

- ✚ **BorderStyle:** estilo de borde de un control.
- ✚ **Button:** control de botón de Windows.
- ✚ **ComboBox:** control de cuadro combinado de Windows.
- ✚ **Chart:** control utilizado para la visualización de las gráficas.
- ✚ **DataGridView:** refleja datos en una cuadrícula personalizable, en este caso tipo Excel.
- ✚ **Form:** ventana o cuadro de diálogo que constituye la interfaz.
- ✚ **HSG.Numerics:** paquete utilizado para la resolución de ecuaciones no lineales.
- ✚ **Label:** etiqueta estándar de Windows.
- ✚ **SerialPort:** control utilizado para la comunicación entre el puerto COM y el PLC.
- ✚ **Textbox:** control de cuadro de texto de Windows.
- ✚ **Timer:** implementa un temporizador que genera un evento en los intervalos definidos por el usuario.

1.4 Tipos de datos utilizados

- ✚ **Array:** representa un vector.
- ✚ **Bool:** valor booleano que puede ser true o false.
- ✚ **Double:** representa un numero decimal.
- ✚ **DataTable:** representa una tabla con DataRow (filas) y DataColumn (columnas).
- ✚ **DateTime:** representa una unidad de tiempo.
- ✚ **String:** representa una cadena de caracteres.
- ✚ **Void:** tipo de valor devuelto para un método que no devuelve un valor.

1.5 Eventos

Los eventos son señales que aparecen en el sistema cuando se está programando, se encarga de dar aviso para que se pueda dar pronta respuesta. Por ejemplo, si el usuario hace clic en el botón de una página web, se puede reaccionar a esa acción y mostrar una tarjeta con información.

Los eventos utilizados en la creación del proyecto se plantean a continuación:

- ✚ **Click:** al “hacer clic” en cualquier control.
- ✚ **TimerTick:** cuando ha transcurrido el intervalo del temporizador especificado y está habilitado.
- ✚ **CheckedChanged:** al momento de seleccionar un RadioButton.
- ✚ **SelectedIndexChanged:** cambio de un ítem en el ComboBox.
- ✚ **SerialPort_DataReceived:** envío de un dato determinado desde el puerto COM.
- ✚ **DisplayText:** recibo de datos del puerto COM hasta **SerialPort_DataReceived**.

2. Creación de un proyecto en Visual Studio Code

Se creará un proyecto de aplicación C#. en el tipo de proyecto se incluyen las diferentes plantillas que se pueden utilizar para su realización.

2.1 Abra Visual Studio Code



Figura 2. Icono del software utilizado.

2.2 En la ventana de inicio, seleccione “crear un proyecto”.

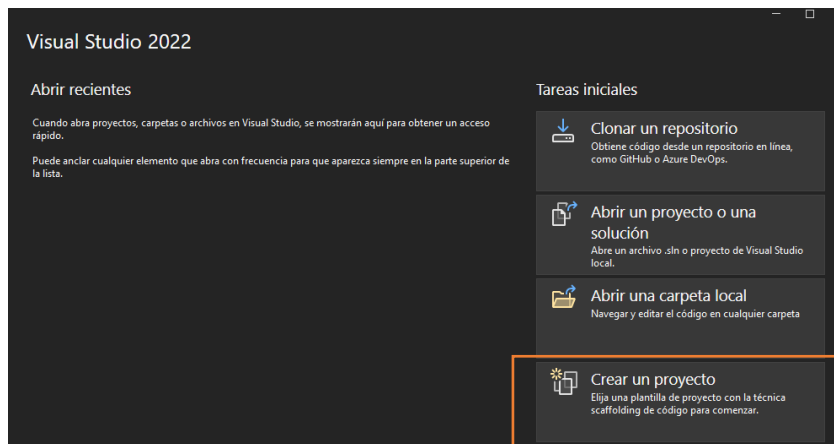


Figura 2.1. Ventana para crear un proyecto.

2.3 Al seleccionar “crear un proyecto”, se escoge la plantilla Windows Forms App (.NET Framework) para C#.

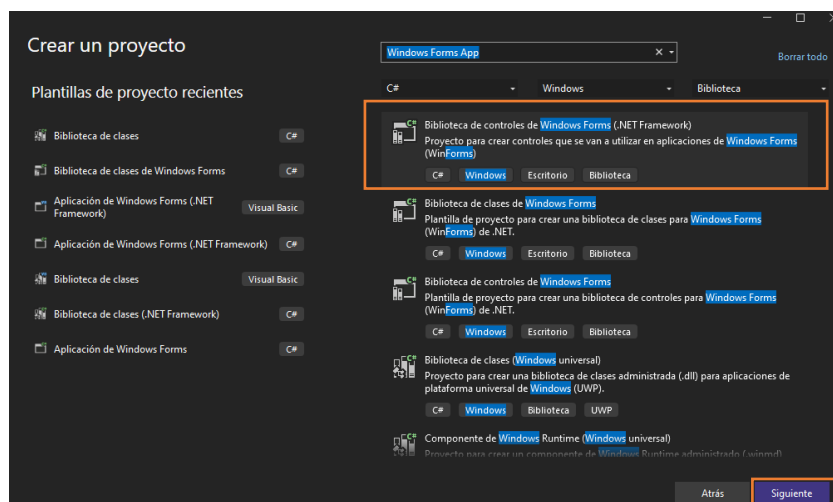


Figura 2.2. Selección de la biblioteca a trabajar.

2.4 Al darle clic en el botón “siguiente” se abrirá una nueva ventana llamada “configure su nuevo proyecto” en donde se escribirá el nombre de este y posteriormente se le dará clic en “crear”.

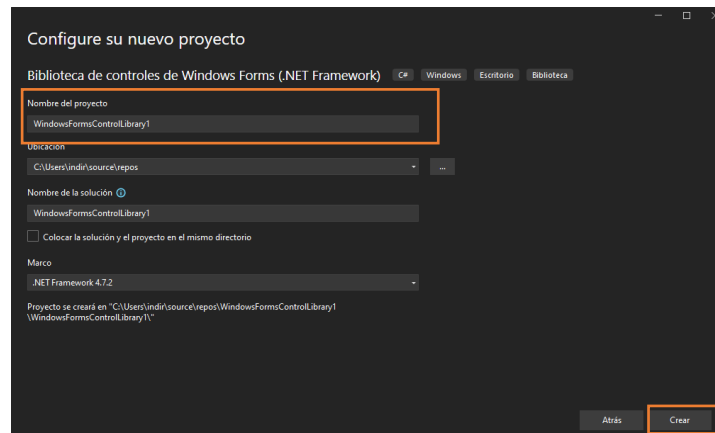


Figura 2.3. Configuración principal del proyecto.

3. Creación de una aplicación

Al seguir los pasos anteriormente mencionados Visual Studio Code abre un formulario de forma automática, este es una interfaz de usuario de Windows en donde se pueden agregar diferentes controles para ejecutarla.

3.1 En el cuadro de herramientas que se encuentra en la parte izquierda del entorno se selecciona la opción de “controles comunes” en donde se encontrarán los diferentes elementos que se estarán utilizando a lo largo del proyecto dependiendo del diseño que se requiera y en la parte derecha el explorador de soluciones el cual es una herramienta que contiene el listado de las bibliotecas de clases.

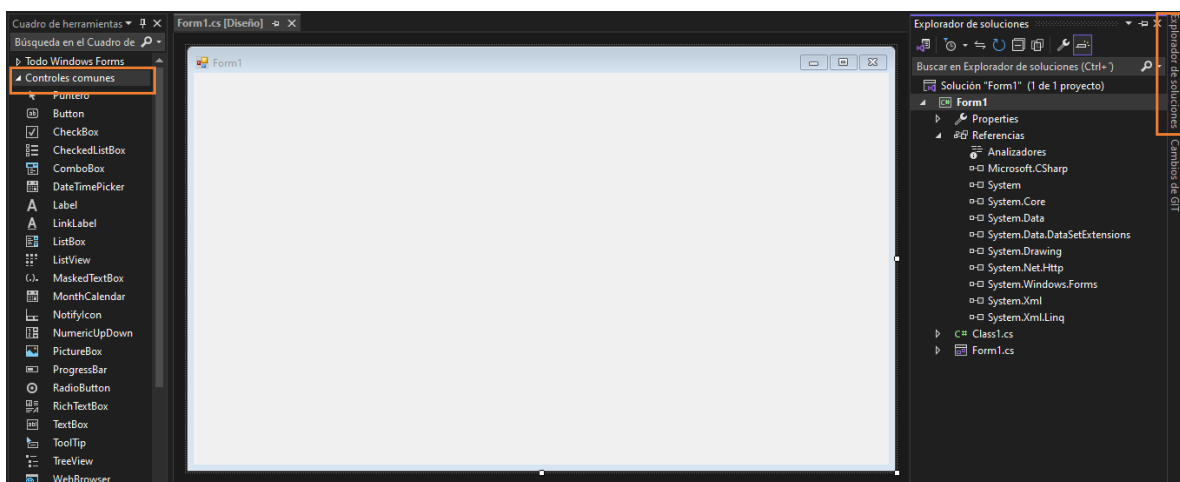


Figura 3. Entorno del formulario.

3.2 Al insertar cualquier control como por ejemplo un botón, al darle doble clic se presenta una serie de propiedades las cuales permiten diseñar cada una de las herramientas de acuerdo a las necesidades que requiera. Aquí se encuentran diferentes tipos de fuentes, tamaño, bordes, márgenes, localización, etc.

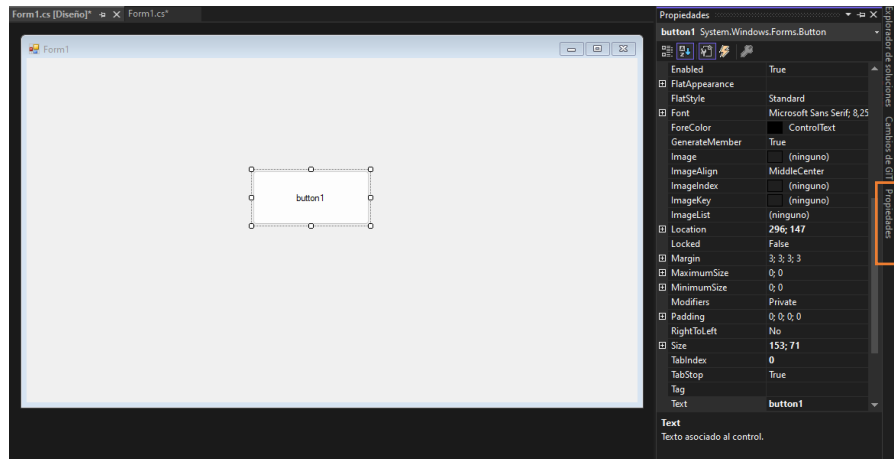


Figura 3.1 Selección de propiedades.

3.3 Al crearse un diseño también se crea una pestaña en donde se puede ver y editar el código fuente para cada uno de los elementos que se inserta en el diseñador.

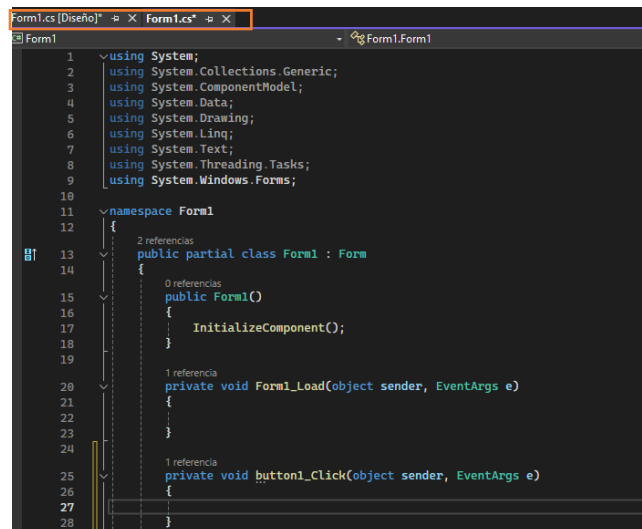


Figura 3.2. Código fuente.

4. Ejecución para el módulo automático

En la aplicación desarrollada se presentan dos modos: manual y automático. De manera manual se ingresan los parámetros iniciales para el estudio como lo son la caída de presión, velocidad lineal del gas y velocidad del flujo de aire (Jsl). El modo automático permite visualizar el valor de la caída de presión gracias a la conexión entre el puerto COM habilitado desde Visual Studio Code y el PLC.

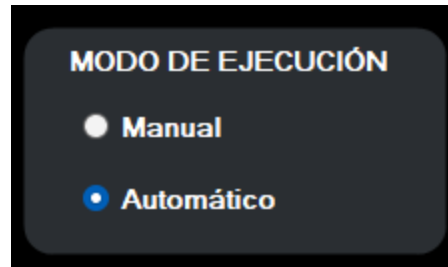


Figura 4. Modos de ejecución.

Al crearse el botón de comando el código correspondiente es el siguiente donde se puede calibrar, inhabilitar o habilitar según sea el caso.

```
private void radioButton1_CheckedChanged(object sender, EventArgs e)
{
    btnCalibrar.Enabled = radioButton1.Checked;
    txtDeltaP.ReadOnly = radioButton1.Checked;
    txtVelLinealGas.ReadOnly = radioButton1.Checked;
    txtVelLinealGas.Clear();
    txtDeltaP.Clear();
}
```

Figura 4.1. Código ejecución modo automático.

Para el modo manual el código correspondiente es el siguiente donde se puede calibrar, inhabilitar o habilitar según sea el caso.

```
private void radioButton2_CheckedChanged(object sender, EventArgs e)
{
    btnCalibrar.Enabled = radioButton1.Checked;
    txtDeltaP.Focus();
}
```

Figura 4.2. Código de ejecución modo manual.

5. Ejecución herramientas

En el panel de herramientas se encuentran diferentes opciones como se muestran a continuación:



Figura 5. Herramientas de la aplicación.

5.1 Nuevos resultados: diseñado para borrar los datos almacenados y guardar los nuevos. Su código fuente se presenta a continuación:

```
foreach (DataRow row in tblConcentracion.Rows)
{
    string serie = cmbParamGraficar.Text + " " + row["concentracion"].ToString();
    grafica.Series.Add(serie);
    int.TryParse(row["concentracion_id"].ToString(), out int concentracion);
    table = resultadoRepository.DiámetroBurbujaVsJg(concentracion, espumante);
    string[] cols = { "jg", "db" };
    ControlForm.GetGraphic(grafica, cmbtipoGrafica.SelectedValue.ToString(), serie, cols, table);
}
break;

string[] arr = [];
table = resultadoRepository.DiámetroBurbujaVsConcentracion(espumante);
ControlForm.FillArray(grafica, table, ref arr);
string[] cols = ["concentracion", "db"];
for (int j = 0; j <= arr.Length - 1; j++)
{
    var search = Table.Busqueda("jg", arr[j], table);
    ControlForm.GetGraphic(grafica, cmbtipoGrafica.SelectedValue.ToString(), arr[j], cols, search);
}
break;
```

Figura 5.1. Código nuevos resultados.

5.2 Mostrar otros resultados: diseñado para mostrar parámetros calculados por medio de las ecuaciones tomadas como referencia para el estudio hidrodinámico:

```
1 referencia
private void button4_Click(object sender, EventArgs e)
{
    frmDatosEntrada frmDatos = new frmDatosEntrada
    {
        dt1 = Utilities.Table.GetTerminos("P1", Utilities.CaidaDePresion.PrimerTermino),
        dt2 = Utilities.Table.GetTerminos("ReynoldEnjambre", Utilities.CaidaDePresion.ReynoldEnjambre),
        dt3 = Utilities.Table.GetTerminos("SegundoTermino", Utilities.CaidaDePresion.SegundoTermino),
        dt4 = Utilities.Table.GetTerminos("TercerTermino", Utilities.CaidaDePresion.TercerTermino),
        dt5 = Utilities.Table.GetTerminos("FuncionObjetivo", Utilities.CaidaDePresion.FuncionObjetivo),
        dt6 = Utilities.Table.GetTerminos("DiametroBurbuja", Utilities.CaidaDePresion.DiametroBurbuja)
    };
    frmDatos.ShowDialog();
}
```

Figura 5.2. Código otros resultados.

5.3 Ver valores iniciales: diseñado para visualizar parámetros iniciales como densidades, gravedad, diámetros, velocidad lineal, etc.

```
private void button3_Click(object sender, EventArgs e)
{
    frmValoresIniciales frmValoresIniciales = new frmValoresIniciales { dt = Utilities.Table.GetInitialValues() };
    frmValoresIniciales.ShowDialog();
}
```

Figura 5.3. Código ver valores iniciales.

5.4 Graficar resultados: diseñado para visualizar las diferentes tablas con las características seleccionadas.

```
1 referencia
private void btnGraficar_Click(object sender, EventArgs e)
{
    // frmGrafica frmGrafica = new frmGrafica();

    double.TryParse(txtDeltaP.Text, out double DeltaP);
    double.TryParse(txtVellinealGas.Text, out double VellinealGas);
    double.TryParse(txtJsl.Text, out double Jsl);

    if (DeltaP == 0)
    {
        MessageBox.Show("Delta p debe ser mayor que 0", "", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }
    if (VellinealGas == 0)
    {
        MessageBox.Show("velocidad lineal gas debe ser mayor que 0", "", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }
    if (Jsl == 0)
    {
        MessageBox.Show("Jsl debe ser mayor que 0", "", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    double reynold = Utilities.CaidaDePresion.CalcularValorReynold(DeltaP, VellinealGas, Jsl);
    string[] columns = { "Variable", "Resultado" };
    DataTable dt = Utilities.Table.GetDataTable(columns);
    string[] values = { "Reynold", reynold.ToString() };
    Utilities.Table.SetRow(dt, columns, values);
    string[] values2 = { "Ub", Utilities.CaidaDePresion.ub.ToString() };
    Utilities.Table.SetRow(dt, columns, values2);
    string[] values3 = { "Hold up", Utilities.CaidaDePresion.holdup.ToString() };
}
```

Figura 5.5. Código graficar resultados.

6. **Timer:** este control ejecutará una sentencia cada determinado tiempo. Aquí se carga la hora del sistema

```

I referencia
private void frmPrincipal_Load(object sender, EventArgs e)
{
    lblReloj.Text = DateTime.Now.ToString("hh:mm:ss");
    timer1.Start();
    tooltip1.SetToolTip(button4, "Mostrar otros resultados");
    tooltip1.SetToolTip(button3, "Ver valores iniciales");
    tooltip1.SetToolTip(btnCalibrar, "Calibrar plc");
    tooltip1.SetToolTip(btnGraficar, "Graficar resultados");
    pnlGraficas.Controls.Add(grafica);
}

```

Figura 6. Código reloj.

Nota: este control tiene una propiedad llamada Interval, la cual es la que controla el intervalo de tiempo de una ejecución, para cambiarla seleccione propiedades de control y digite el numero en milisegundo en cual quiere que se ejecute cada código.

7. Insertar imágenes

Para añadir imágenes, seleccionar en cuadro de herramienta el control "picturebox", luego dirigirse a propiedades en la cual se seleccionará "image" haga clic sobre (...) y aparecerá el cuadro de dialogo abrir donde podrá seleccionar la imagen deseada.



Figura 7. Insertar imagen.

Para añadir el logo seleccionar en cuadro de herramienta el control “panel” luego dirijase a propiedades y seleccionar “backgroundimage” haga clic sobre (...) y aparecerá el cuadro de dialogo donde podrá seleccionar la imagen que desee; en la propiedad “backgroundimagelayout” seleccione el “item stretch”.



Figura 7.1. Insertar logo.

8. Ejecución de la aplicación

Para ejecutar la aplicación en Visual Studio Code, seleccionar en la barra de herramientas el icono “play”; en este caso aparecerá el nombre del proyecto “caída de presión” acompañado de dicho icono. Al seleccionarlo el programa se abrirá en una nueva ventana donde se encontrará el diseño realizado.

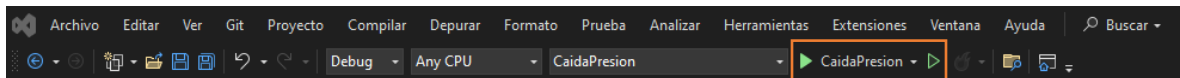
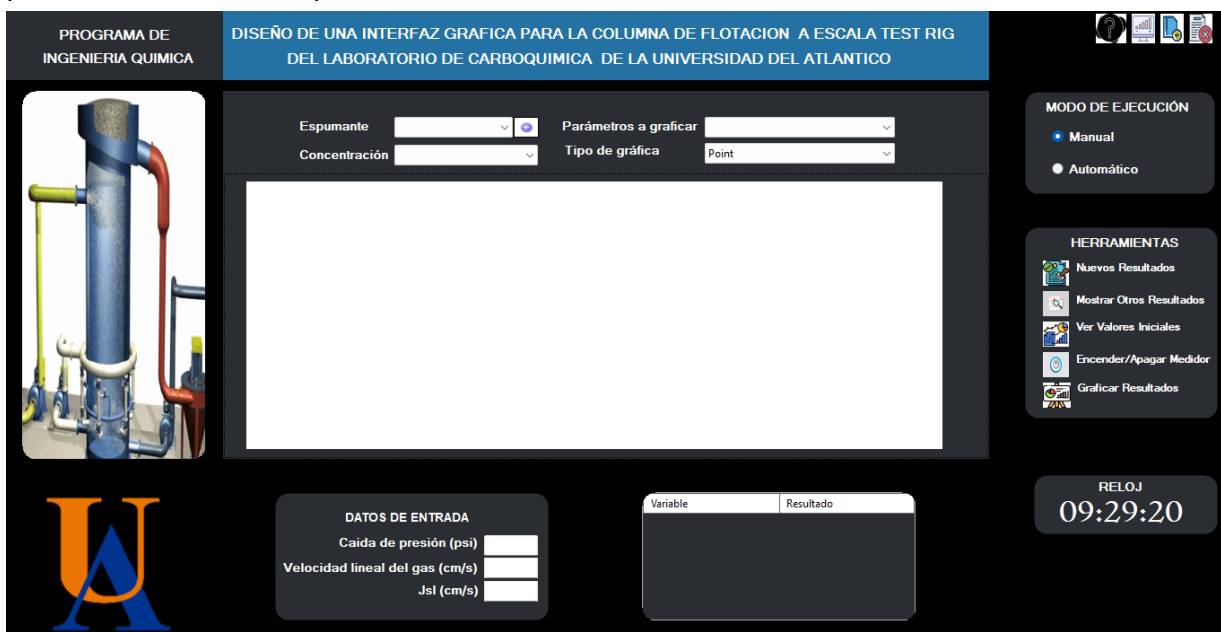


Figura 8. Vista general de la aplicación.

El diseño se presentará en una nueva pantalla donde se podrán ingresar todos los parámetros necesarios para llevar a cabo el análisis hidrodinámico.



9. Almacenamiento de datos

Luego de ingresar cada uno de los parámetros requeridos para el estudio hidrodinámico, la aplicación guardará cada uno de esos datos por medio de MySQL. Es importante aclarar que las bases de datos son el repositorio de datos esencial para todas las aplicaciones de software.

Aquí se almacena los datos en tablas separadas en lugar de poner todos los datos en un gran almacén. El modelo de datos lógico, con objetos como tablas de datos, vistas, filas y columnas, ofrece un entorno de programación flexible.

Esta base de datos es completamente libre y es compatible con muchos lenguajes de programación entre ellos Visual Studio Code versión community.

A continuación, se presenta la forma en la que está organizada la base de datos en la aplicación para la columna de flotación.

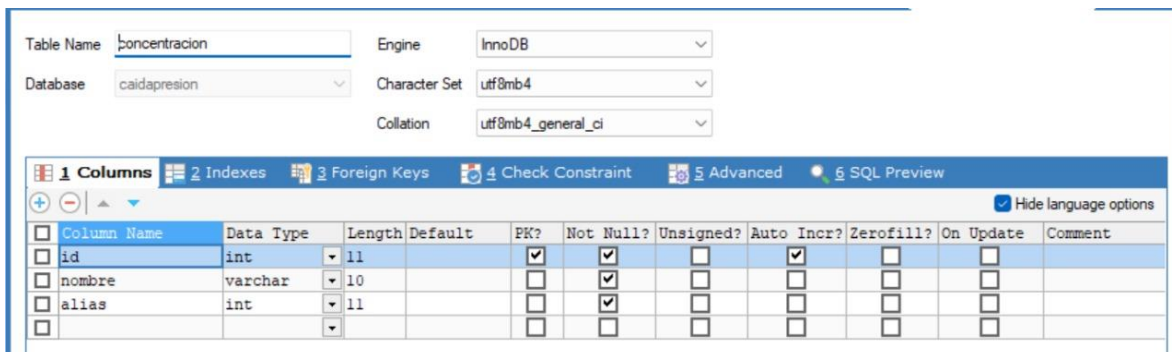


Table Name: concentracion Engine: InnoDB
Database: caidapresion Character Set: utf8mb4
Collation: utf8mb4_general_ci

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update	Comment
<input type="checkbox"/> id	int	11		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> nombre	varchar	10		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> alias	int	11		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 9. Base de datos creada para las concentraciones.

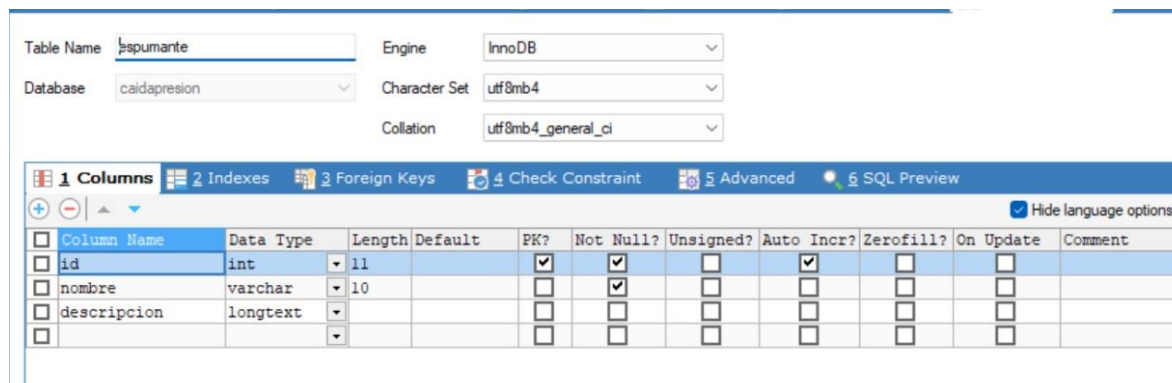


Table Name: espumante Engine: InnoDB
Database: caidapresion Character Set: utf8mb4
Collation: utf8mb4_general_ci

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update	Comment
<input type="checkbox"/> id	int	11		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> nombre	varchar	10		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> descripcion	longtext			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 9.1. Base de datos creada para los espumantes.

Table Name: grafica Engine: InnoDB

Database: caidapresion Character Set: utf8mb4

Collation: utf8mb4_general_ci

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update	Comment
<input type="checkbox"/> id	int	11		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> nombre	varchar	50		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 9.2. Base de datos creada para las diferentes gráficas.

Table Name: otros_resultados Engine: InnoDB

Database: caidapresion Character Set: utf8mb4

Collation: utf8mb4_general_ci

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update	Comment
<input type="checkbox"/> id	int	11		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> PrimerTermino	decimal	15,15		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> ReynoldEnjambre	decimal	15,15		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> SegundoTermino	decimal	15,15		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> TercerTermino	decimal	15,15		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> FuncionObjetivo	decimal	15,15		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> DiametroBurbuja	decimal	15,15		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> resultado_id	int	11		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 9.3 Base de datos creada para mostrar otros resultados.

Table Name: resultados Engine: InnoDB

Database: caidapresion Character Set: utf8mb4

Collation: utf8mb4_general_ci

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	On Update	Comment
<input type="checkbox"/> id	int	11		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> deltap	decimal	10,2		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> js1	decimal	10,2		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> holdup	decimal	10,2		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> db	decimal	10,2		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> ub	decimal	10,2		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> reb	decimal	10,2		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> usg	decimal	10,2		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> jg	decimal	10,2		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> concentracion_id	int	11		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/> espumante_id	int	11		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 9.4 Base de datos creada para los resultados principales.

En el momento que se encuentre en el entorno de la base de datos en MySQL, se encontrará con diferentes columnas en las cuales deberá escribir lo que se requiera; por ejemplo en "COLUMN NAME" se escribirá el nombre de la variable que se almacenará, "DATA TYPE" hace énfasis en el tipo de dato sea entero, decimal o un dato de longitud indeterminada, "LENGTH" es el tamaño del archivo, "PK" es lo que nos permite conectar con el valor, "NOT NULL" al momento de seleccionar esta casilla indicamos que no aceptamos valores nulos, "AUTO INCR" ayuda a incrementar automáticamente el valor.