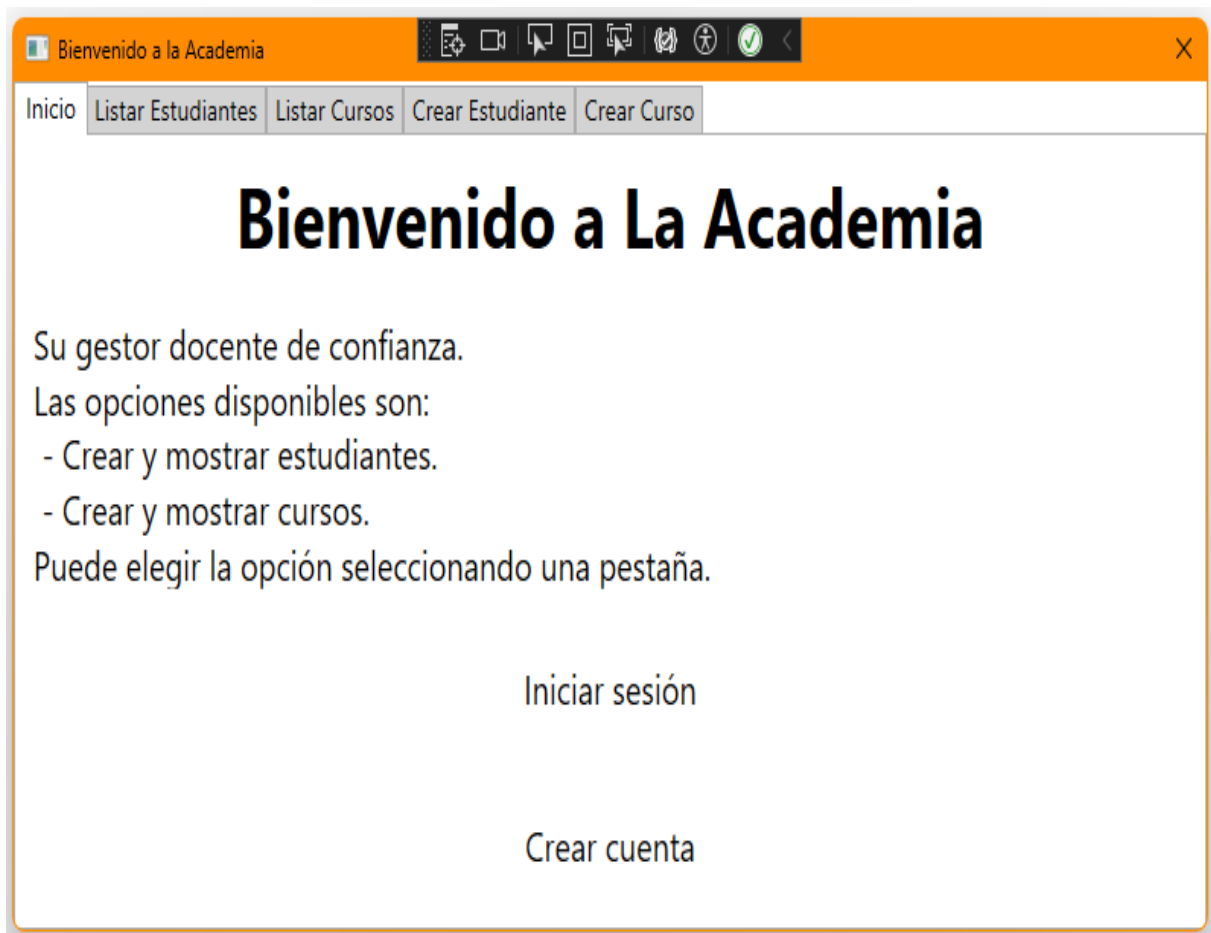


# Academia



Img.1 - Pantalla principal de Academia al iniciar.

## Índice

0. Portada	1
1. Estructura de la app	3
2. Funcionamiento y diseño	4
2.1. Funcionamiento	5
2.2. Diseño	10
3. Decisiones	10
4. Posibles mejoras	11

## Estructura de la app

### Clases, ventanas y páginas:

#### Clases:

La estructura de la aplicación está dividida por funcionalidad, utilizando un patrón de arquitectura MVVM (Model-View-ViewModel, o Modelo-Vista-ModeloDeVista) que facilita la actualización de la Interfaz de Usuario mediante un ViewModel compartido. Podemos ver fácilmente las tres capas que hay: UI (sin carpeta, ya que es la mayoría de la aplicación: ventanas y páginas), Data (que contiene los modelos de datos: Student y Course) y el ViewModel (que es el encargado de recibir y distribuir los datos entre las distintas UI).

1. **Course:** Representa un Curso y tiene como atributos CourseName, Duration, ProfessorName y una lista de Student.
2. **Student:** Representa un Estudiante y tiene como atributos FirstName, LastName, Age y una lista de Course.
3. **ViewModel:** Es el ViewModel de la aplicación que se inicializa en la MainWindow y se comparte entre las vistas para mantener la información de las mismas actualizada: p.e., cuando se añade un Curso o un Estudiante, éste se ve reflejado en su lista (en la pestaña correspondiente).

En cuanto a las diferentes vistas (únicamente hay una ventana principal y el resto se presentan en forma de páginas "Page" mediante un TabControl que muestra las diferentes Page dentro de un Frame contenido en su propio TabItem. Básicamente, la ventana MainWindow contiene un TabControl con TabItems que contienen cada uno un Frame que contiene cada página mediante su atributo Source. Así, por ejemplo, el primer TabItem tiene el Header "Inicio" y emplea el Source "Home.xaml" para mostrar la página de inicio con la información relativa a la app. La estructura es la que sigue.

## **Vistas:**

1. **MainWindow**: Se trata de la ventana principal que contiene el TabControl en que se muestran el resto de vistas. En su código (.cs) se instancia también el ViewModel y se pasa a cada una de las vistas para compartir los datos entre las mismas.
2. **ListStudents**: Se trata de la primera Page que se muestra, en la que se listan los estudiantes contenidos en la ObservableCollection Students que se encuentra en el ViewModel. Se muestra mediante un DataGrid utilizando el atributo ItemSource del mismo.
3. **ListCourses**: Se trata de la segunda Page que se muestra, en la que se listan los cursos contenidos en la ObservableCollection Courses que se encuentra en el ViewModel. Al igual que los Student, se muestra mediante un DataGrid utilizando el atributo ItemSource del mismo.
4. **CreateStudent**: Se trata de la tercera Page que se muestra, en la que tenemos tres TextBox que nos permiten introducir los datos del Estudiante a crear. Contiene también dos botones (Aceptar, que guarda el Estudiante si los datos son correctos; y Cancelar que borra todos los TextBox). Se comprueba que la Edad sea un número (únicamente dígitos).
5. **CreateCourse**: Se trata de la cuarta Page que se muestra, en la que tenemos tres TextBox que nos permiten introducir los datos del Curso a crear. Contiene también dos botones (Aceptar, que guarda el Curso si los datos son correctos; y Cancelar que borra todos los TextBox). Se comprueba que la Duración sea un número (únicamente dígitos).

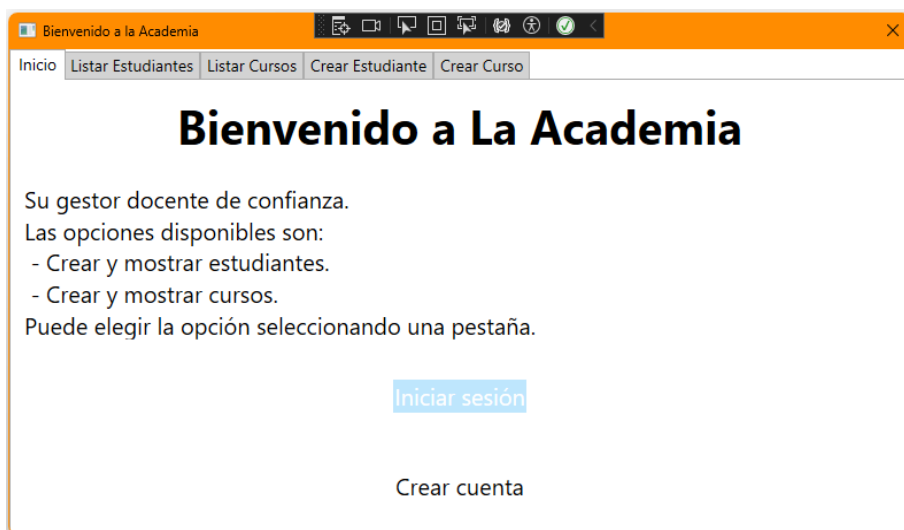
## ***Funcionamiento y diseño***

La aplicación consta de los elementos descritos anteriormente en cuanto a funcionamiento y diseño. El funcionamiento es bastante intuitivo ya que a simple vista tenemos las cinco opciones: Inicio, Listar Estudiantes, Listar Cursos, Crear Estudiante, Crear Curso. Se ha intentado emplear un diseño sencillo, minimizando los errores y la sobrecarga visual.

## Funcionamiento

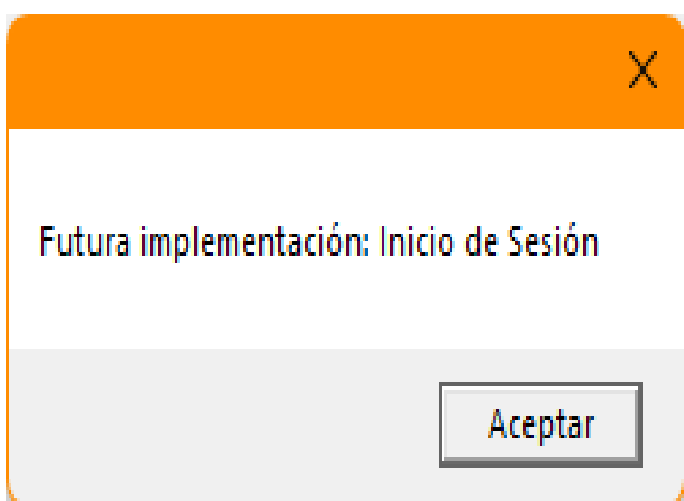
### Inicio

En primer lugar tenemos la página de Inicio, que nos muestra información relativa a la app, un (falso) Inicio de sesión y una (falsa) Creación de cuenta. Ambos con un efecto Hover-over. Al pasar el ratón por encima cambiará el color de los botones y de sus letras, y al dejar de estar encima, éstos volverán a su estado original.



Img. 2 – Efecto Hover-over en “Iniciar sesión”

Al hacer click en Iniciar sesión o en Crear cuenta, se mostrará un MessageBox como se muestra en la imagen 3.



Img. 3 – Click en “Iniciar sesión”

## Listar Estudiantes

Al hacer click en la pestaña “Listar Estudiantes”, se mostrará un listado de todos los Estudiantes (los que se ven inicialmente son ejemplos, ya que se ha poblado inicialmente la lista Students para poder probar la aplicación). Y cuando añadamos un Estudiante usando la pestaña “Crear Estudiante”, al volver a la lista, veremos los datos actualizados.

Nombre	Apellido	Edad
Ernesto	Anastasio	27
Albus	Dumbledore	100
Jesu	Cristus	33
R2	D2	18
Obi-Wan	Kenobi	57

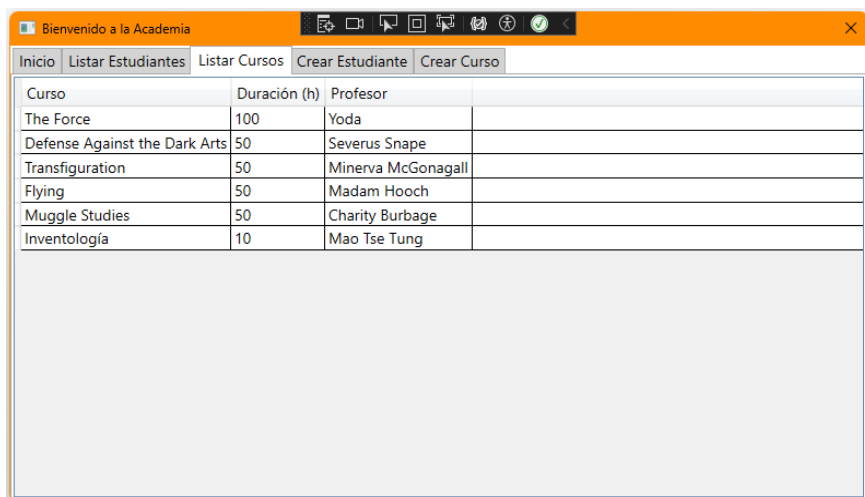
Img. 4 – Pestaña “Listar Estudiantes”

Nombre	Apellido	Edad
Ernesto	Anastasio	27
Albus	Dumbledore	100
Jesu	Cristus	33
R2	D2	18
Obi-Wan	Kenobi	57
Nuevo	Estudiante	23

Img. 5 – Listado actualizado de Estudiantes tras añadir a “Nuevo Estudiante”

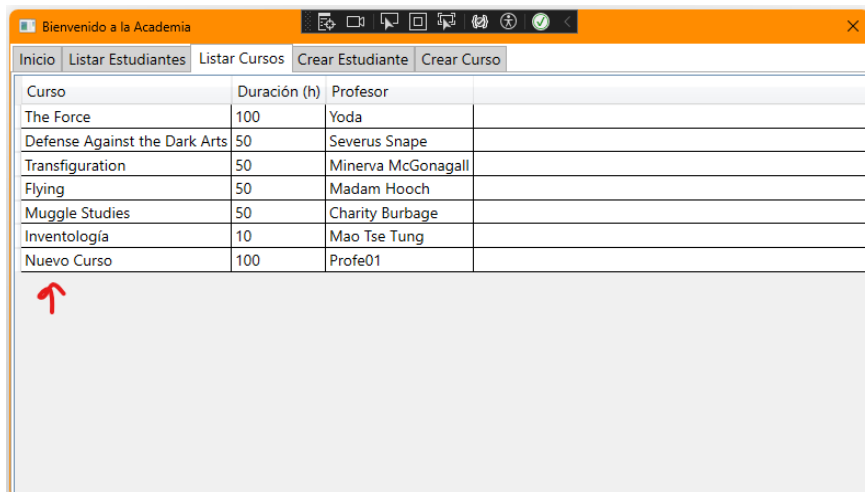
## Listar Cursos

Al hacer click en la pestaña “Listar Cursos”, se mostrará un listado de todos los Cursos (los que se ven inicialmente son ejemplos, ya que se ha poblado inicialmente la lista Courses para poder probar la aplicación). Y cuando añadamos un Curso usando la pestaña “Crear Curso”, al volver a la lista, veremos los datos actualizados.



Curso	Duración (h)	Profesor
The Force	100	Yoda
Defense Against the Dark Arts	50	Severus Snape
Transfiguration	50	Minerva McGonagall
Flying	50	Madam Hooch
Muggle Studies	50	Charity Burbage
Inventología	10	Mao Tse Tung

Img. 6 – Pestaña “Listar Cursos”

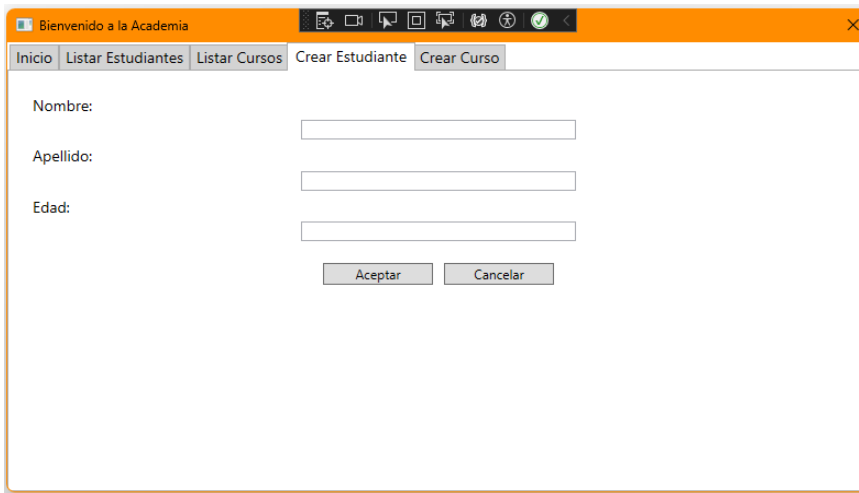


Curso	Duración (h)	Profesor
The Force	100	Yoda
Defense Against the Dark Arts	50	Severus Snape
Transfiguration	50	Minerva McGonagall
Flying	50	Madam Hooch
Muggle Studies	50	Charity Burbage
Inventología	10	Mao Tse Tung
Nuevo Curso	100	Profe01

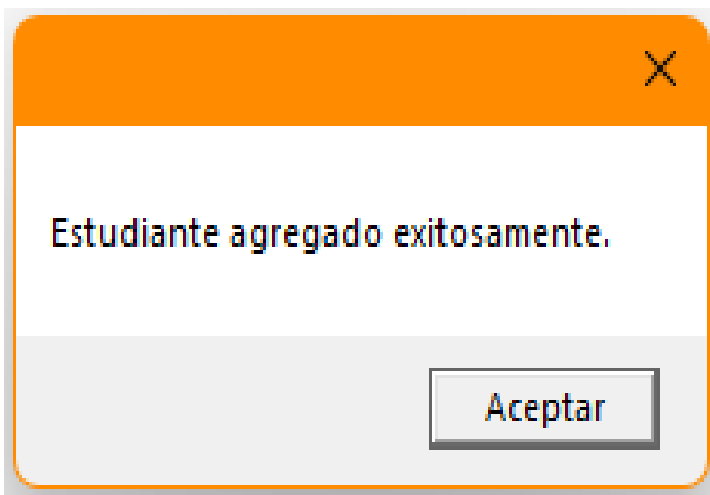
Img. 7 – Listado actualizado de Cursos tras añadir “Nuevo Curso”

## Crear Estudiante

Al hacer click en la pestaña “Crear Estudiante”, se mostrará una página con tres TextBox para introducir los datos del Estudiante y dos Button para Aceptar (añadirlo a la lista Students si los datos son correctos) o Cancelar (limpiar los campos). Tras añadir el Estudiante correctamente se muestra un MessageBox con el mensaje “Estudiante agregado exitosamente”. Tras cerrar el mensaje, los campos se limpiarán para poder añadir otro Estudiante. En caso de que algún campo quede vacío se mostrará un mensaje indicando la obligatoriedad de rellenar todos. Del mismo modo, en caso de que la Edad no sea únicamente un número, se mostrará un mensaje indicándolo.

The image shows a web application window titled "Bienvenido a la Academia". It has a navigation bar with five tabs: "Inicio", "Listar Estudiantes", "Listar Cursos", "Crear Estudiante" (which is active), and "Crear Curso". The main content area of the "Crear Estudiante" tab contains three text input fields labeled "Nombre:", "Apellido:", and "Edad:". Below these fields are two buttons: "Aceptar" and "Cancelar".

Img. 8 – Pestaña “Crear Estudiante”.

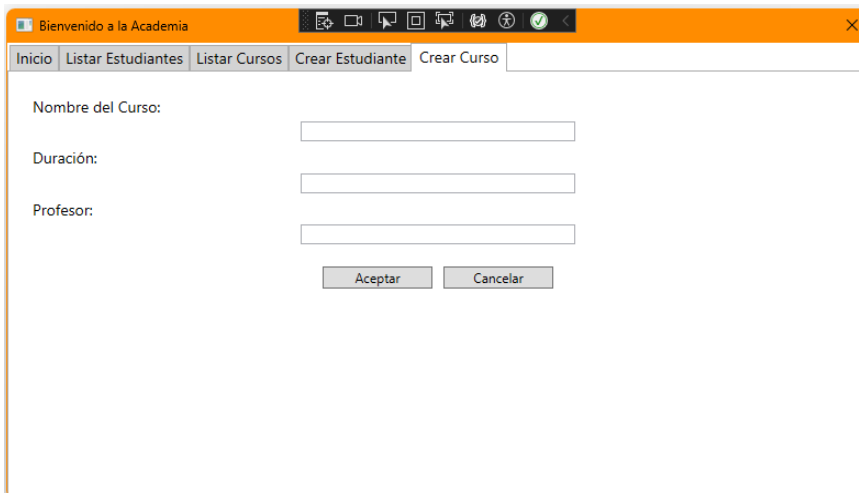


Img. 9 – Mensaje mostrado tras añadir un Estudiante correctamente.



## Crear Curso

Al hacer click en la pestaña “Crear Curso”, se mostrará una página con tres TextBox para introducir los datos del Curso y dos Button para Aceptar (añadirlo a la lista Courses si los datos son correctos) o Cancelar (limpiar los campos). Tras añadir el Curso correctamente se muestra un MessageBox con el mensaje “Curso agregado exitosamente”. Tras cerrar el mensaje, los campos se limpiarán para poder añadir otro Curso. En caso de que algún campo quede vacío se mostrará un mensaje indicando la obligatoriedad de rellenar todos. Del mismo modo, en caso de que la Duración no sea únicamente un número, se mostrará un mensaje indicándolo.

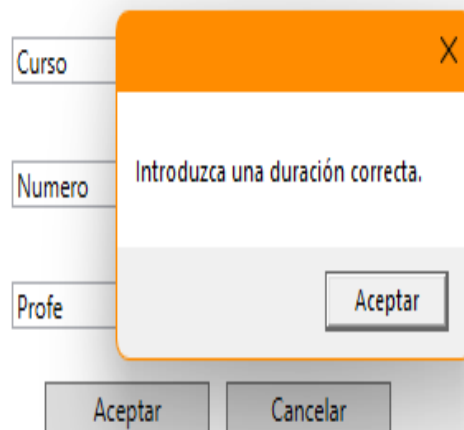


Img. 10 – Pestaña “Crear Curso”.

Nombre del Curso:

Duración:

Profesor:



Img. 11 - Mensaje indicando que Duración debe ser un número.

## Diseño

Se ha tratado de lograr que la interfaz sea simple y compacta, ya que la lógica no es especialmente compleja y las opciones son bastante limitadas. El diseño se ha podido observar a lo largo de todas las capturas y no merece mayor explicación.

## Decisiones

A lo largo del proyecto nos encontramos con diversas decisiones que tomar en cuanto a componentes seleccionados, lógica implementada o incluso el nombres de las variables.

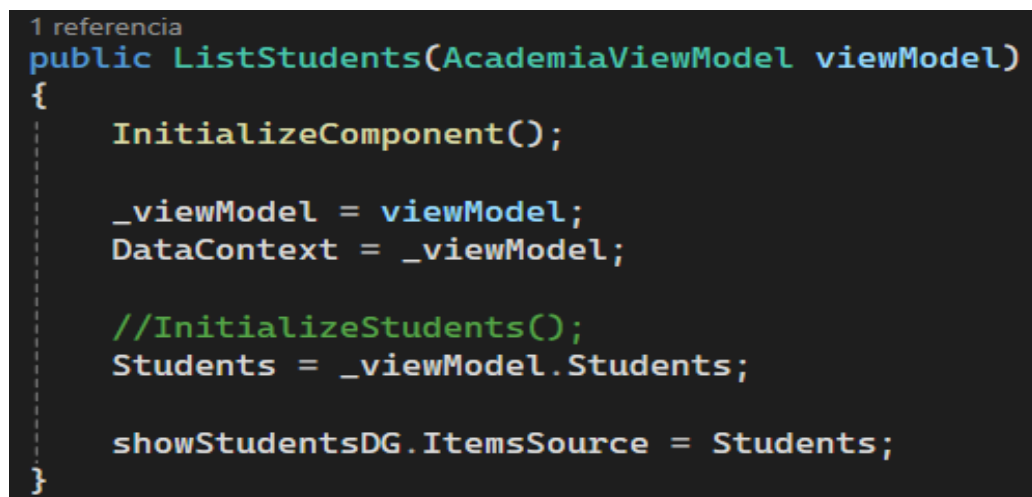
La vinculación de los datos se realiza en cada componente (mediante Binding) como se puede observar en, por ejemplo, la página ListStudents en la que se listan los Estudiantes. Vinculando cada atributo de la clase Student con una columna conseguimos que se muestre cada atributo de cada Student en una columna (y cada Student será, por tanto, una fila). Podemos observar el código en la img. 12.

```
<Page x:Class="Academia.ListStudents"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:local="clr-namespace:Academia"
    mc:Ignorable="d"
    d:DesignHeight="450" d:DesignWidth="800"
    Title="ListStudents">
    <Grid>
        <DataGrid x:Name="showStudentsDG" FontSize="14" AutoGenerateColumns="False">
            <DataGrid.Columns>
                <DataGridTextColumn Header="Nombre" Binding="{Binding FirstName}"/>
                <DataGridTextColumn Header="Apellido" Binding="{Binding LastName}"/>
                <DataGridTextColumn Header="Edad" Binding="{Binding Age}"/>
            </DataGrid.Columns>
        </DataGrid>
    </Grid>
</Page>
```

Img. 12 – DataBinding en ListStudents.xaml

Para que todo ello funcione, es necesario añadir Students como ItemSource de nuestro componente, como se puede apreciar en la img. 13.

**Nota:** como no se vincula la lista de Course asociada a cada Student, no se muestra en la vista.



```
1 referencia
public ListStudents(AcademiaViewModel viewModel)
{
    InitializeComponent();

    _viewModel = viewModel;
    DataContext = _viewModel;

    //InitializeStudents();
    Students = _viewModel.Students;

    showStudentsDG.ItemsSource = Students;
}
```

Img. 13 – DataBinding en ListStudents.xaml.cs

### ***Posibles mejoras***

Añadir funcionalidades como eliminar Estudiantes y Cursos.

Implementar una ventana para el Inicio de sesión antes de poder acceder al resto de opciones.

Implementar una ventana para la Creación de cuentas que posteriormente nos permita iniciar sesión con la misma.

Añadir una foto a cada Estudiante.