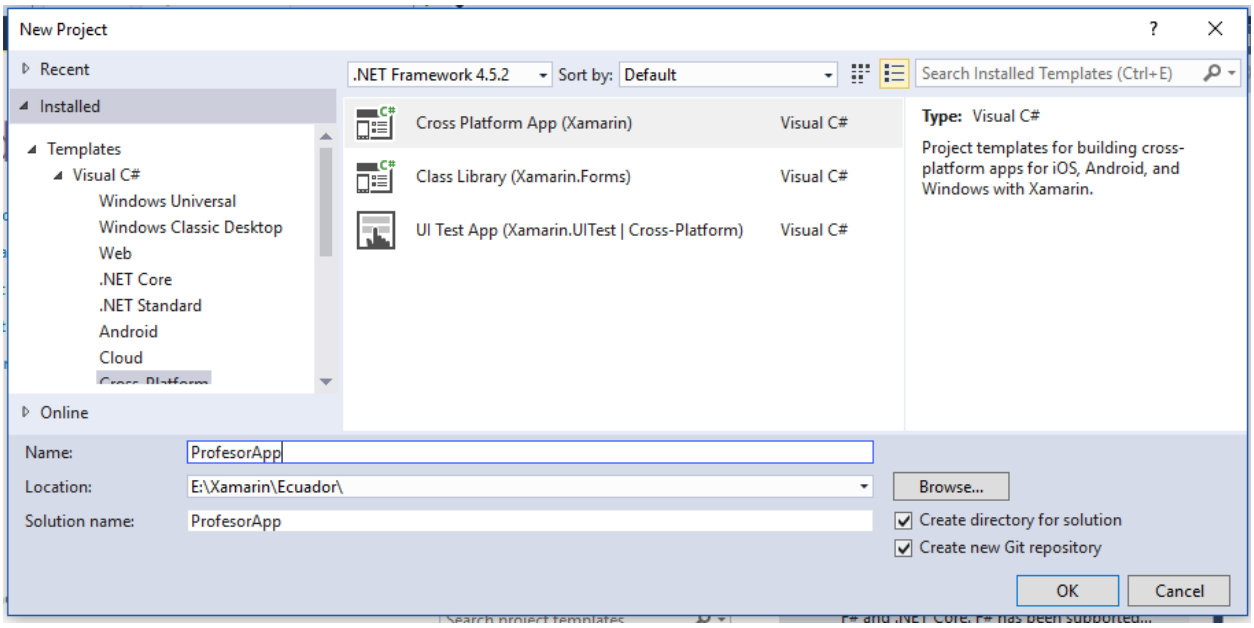
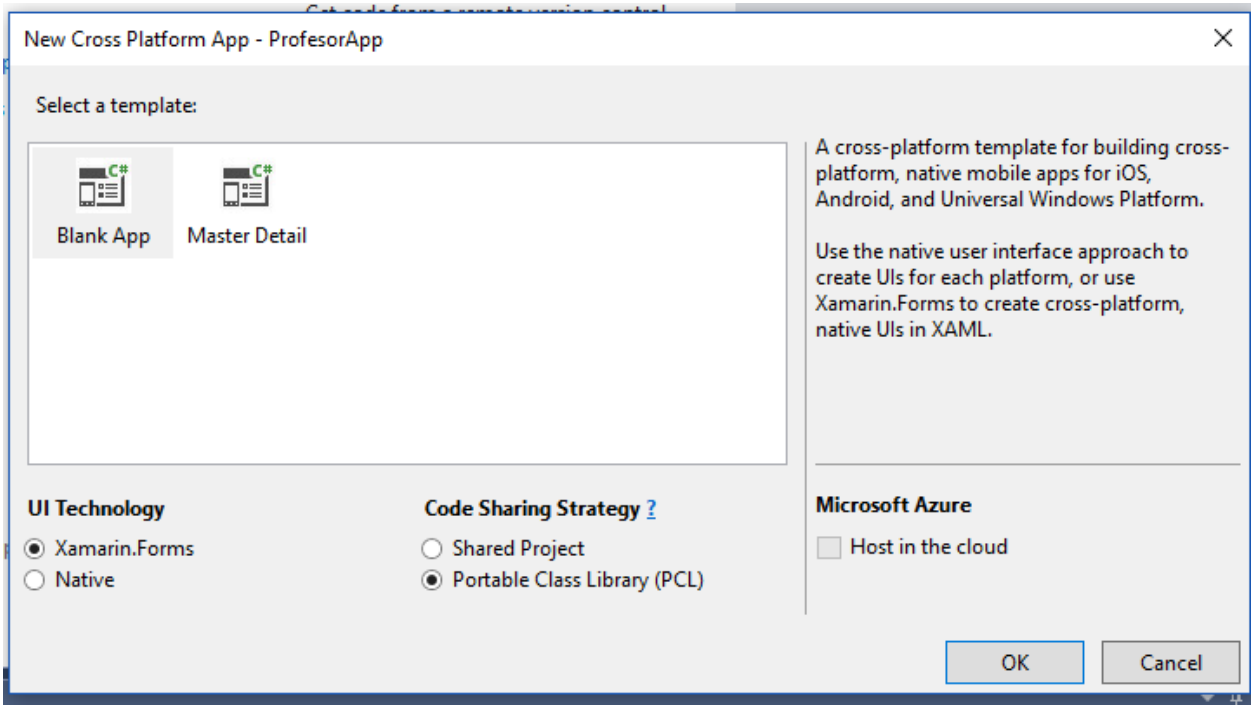
Parte 4: Creación de la app móvil ProfesorApp – Autor: Luis Beltrán

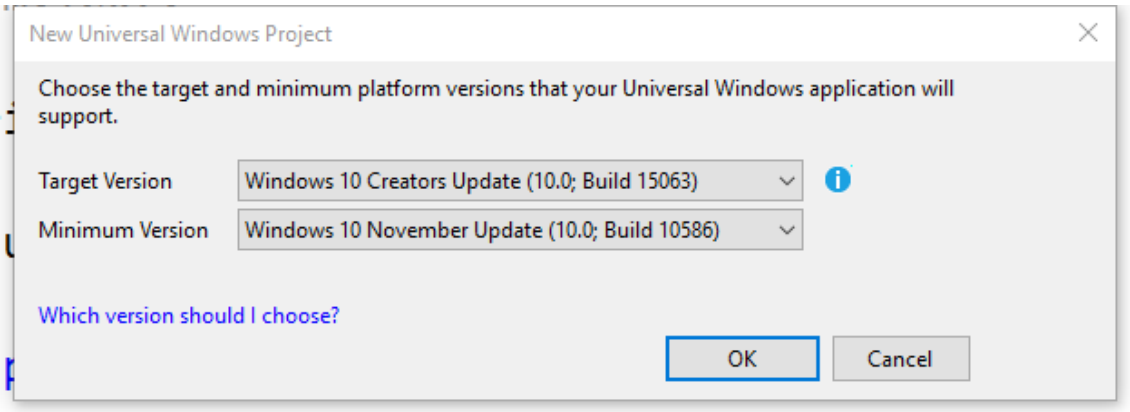
Paso 1. Crea un nuevo proyecto de la categoría **Cross-Platform** selecciona **Aplicación multiplataforma (Xamarin.Forms o nativa)** y coloca el nombre de proyecto **ProfesorApp**. Además, la ruta del proyecto debe ser una ubicación corta para evitar problemas de ruta larga.



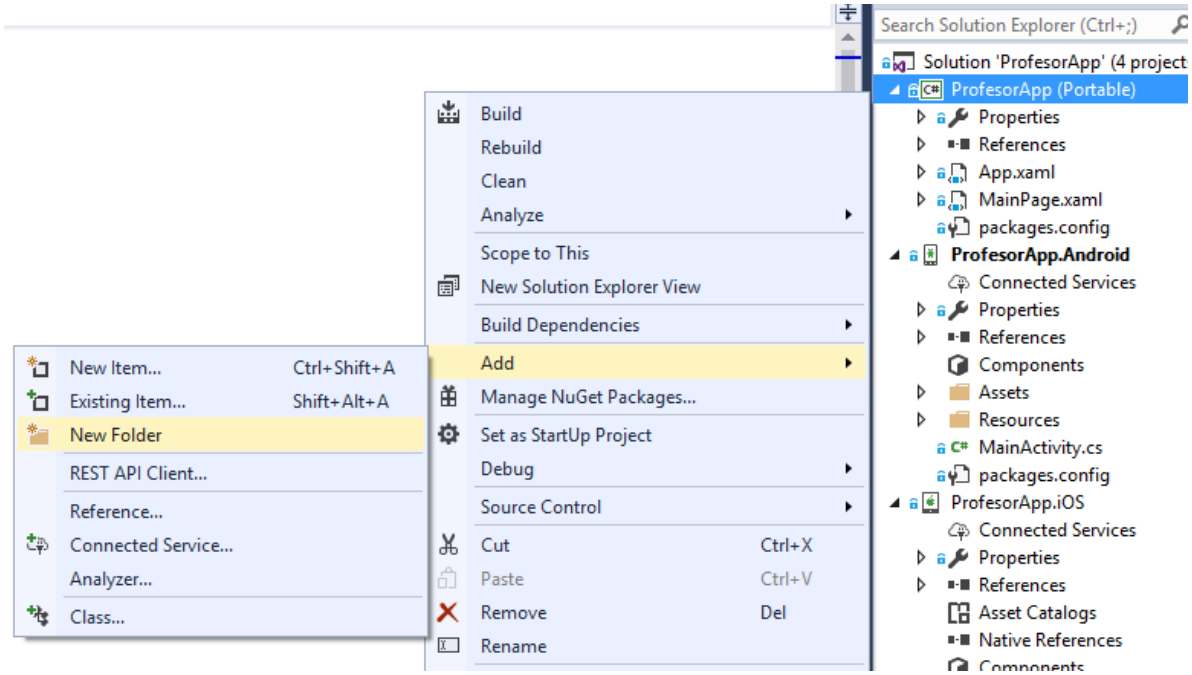
Paso 2. Selecciona la plantilla **Aplicación en blanco**, la tecnología de IU **Xamarin.Forms** y la estrategia de uso compartido de código **Biblioteca de clases portátil (PCL)**. Da clic en **OK**.



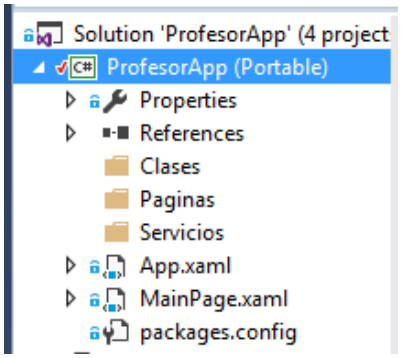
Paso 3. Si tienes instalado el SDK de Windows 10, aparecerá la ventana de selección del Target y Minimum Version. Selecciónalas a conveniencia, según la versión que tengas instalada.



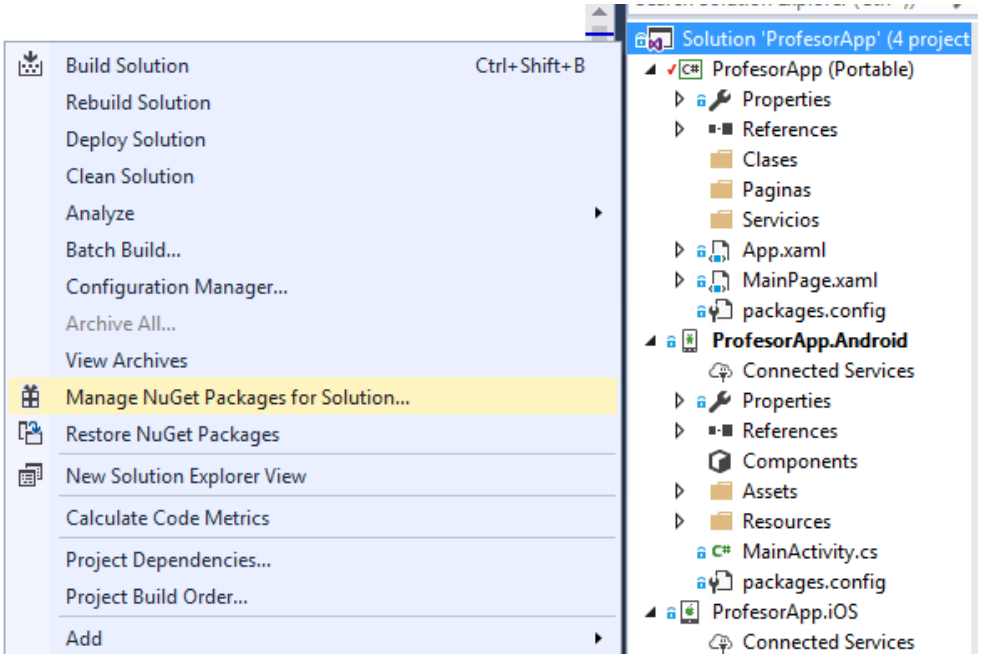
Paso 4. Da clic en **Agregar → Nueva carpeta** en el menú contextual del proyecto PCL.



Paso 5. Agrega las carpetas Clases, Paginas y Servicios al proyecto

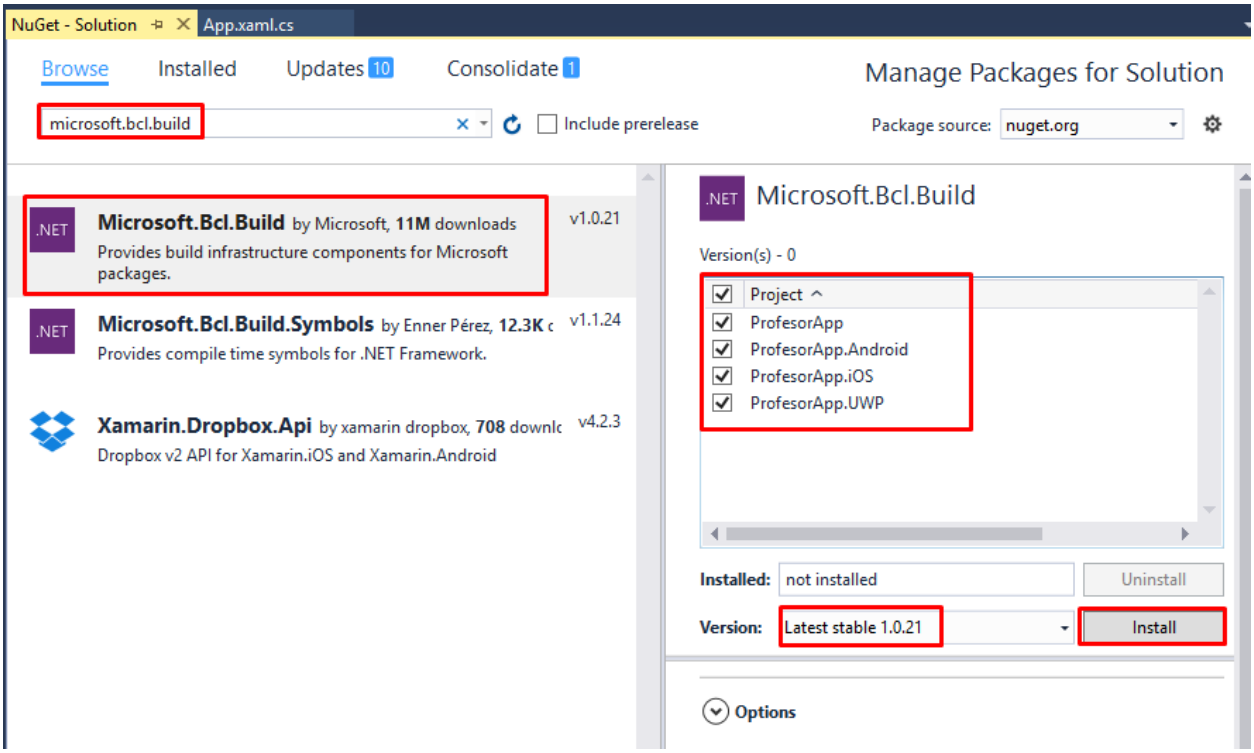


Paso 6. Da clic en **Administrar paquetes Nuget para la solución** en el menú contextual de la solución.

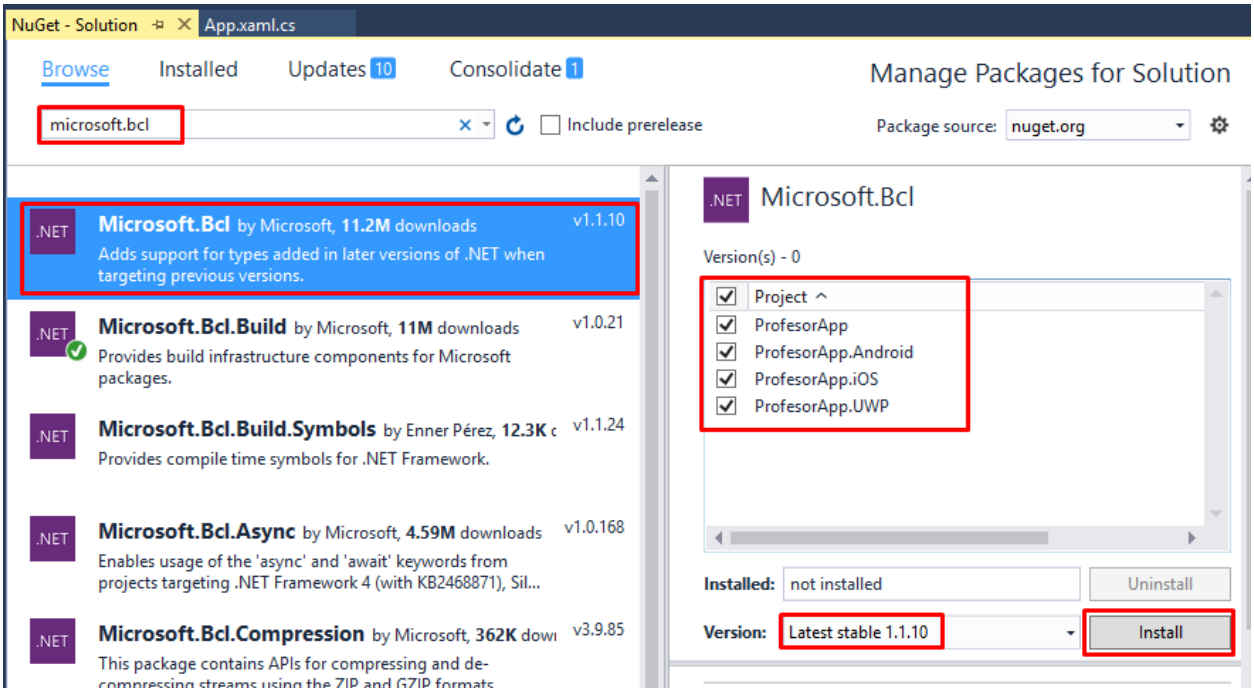


Paso 7. Agrega los siguientes paquetes. Revisa la versión en cada caso:

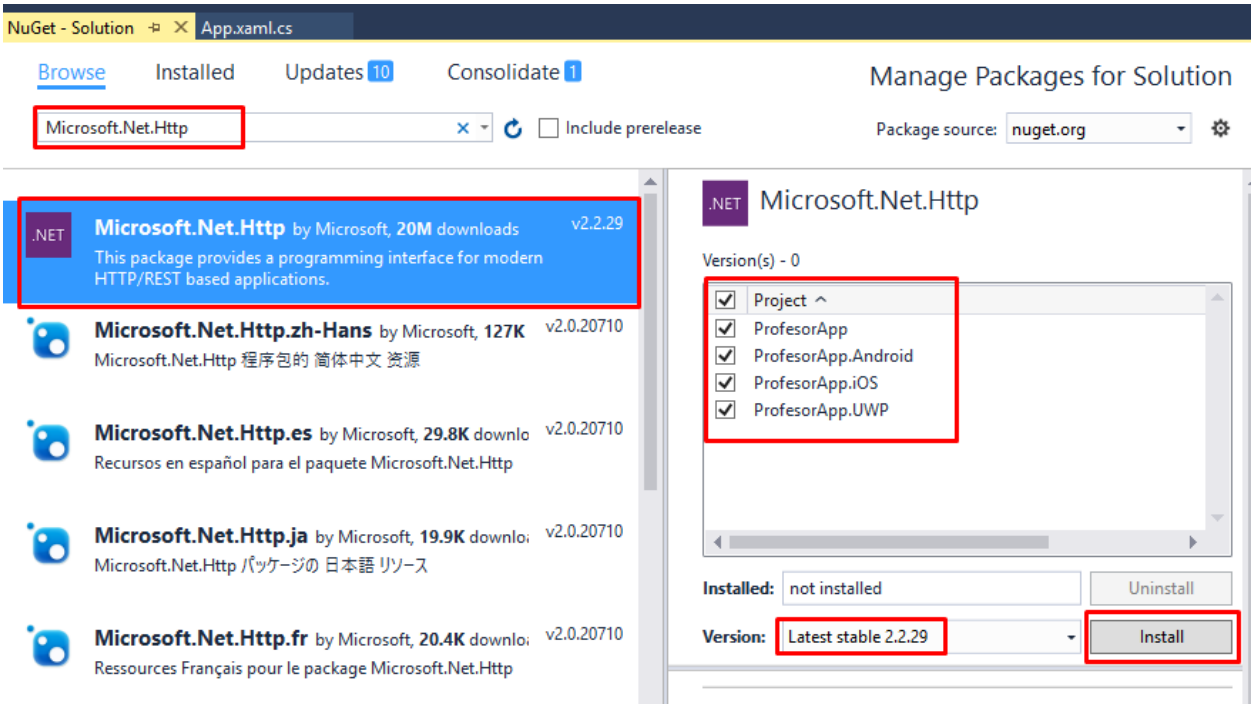
a) Microsoft.Bcl.Build



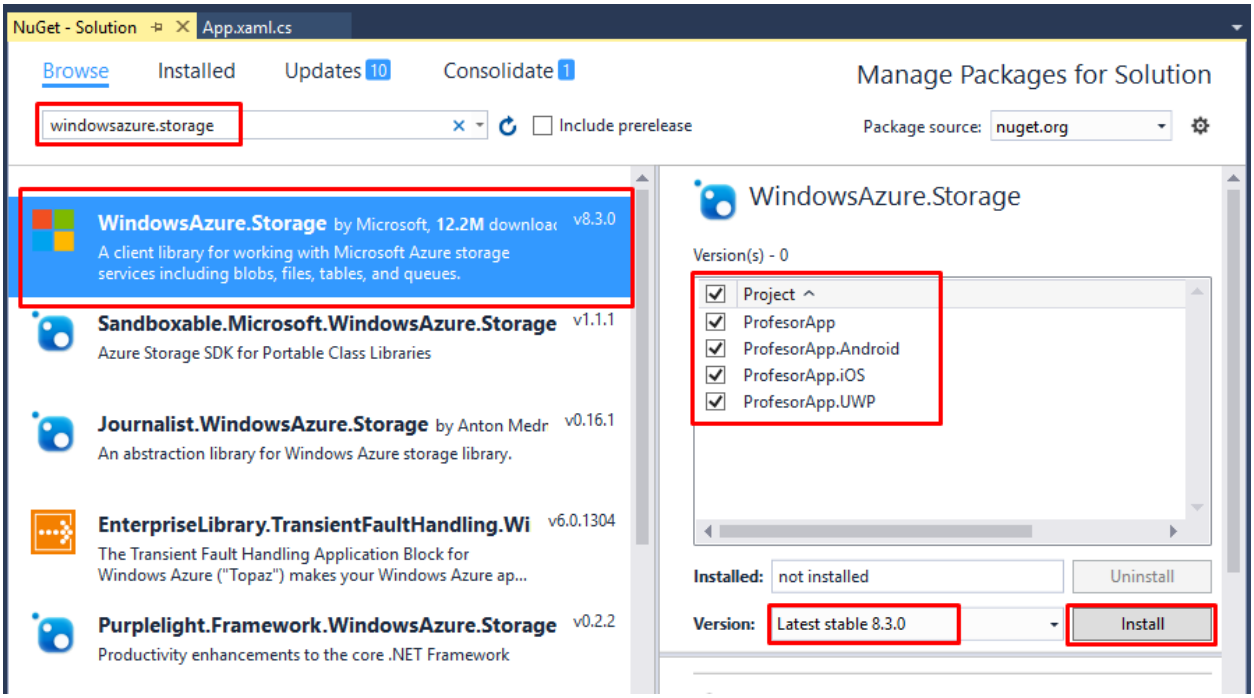
b) Microsoft.Bcl



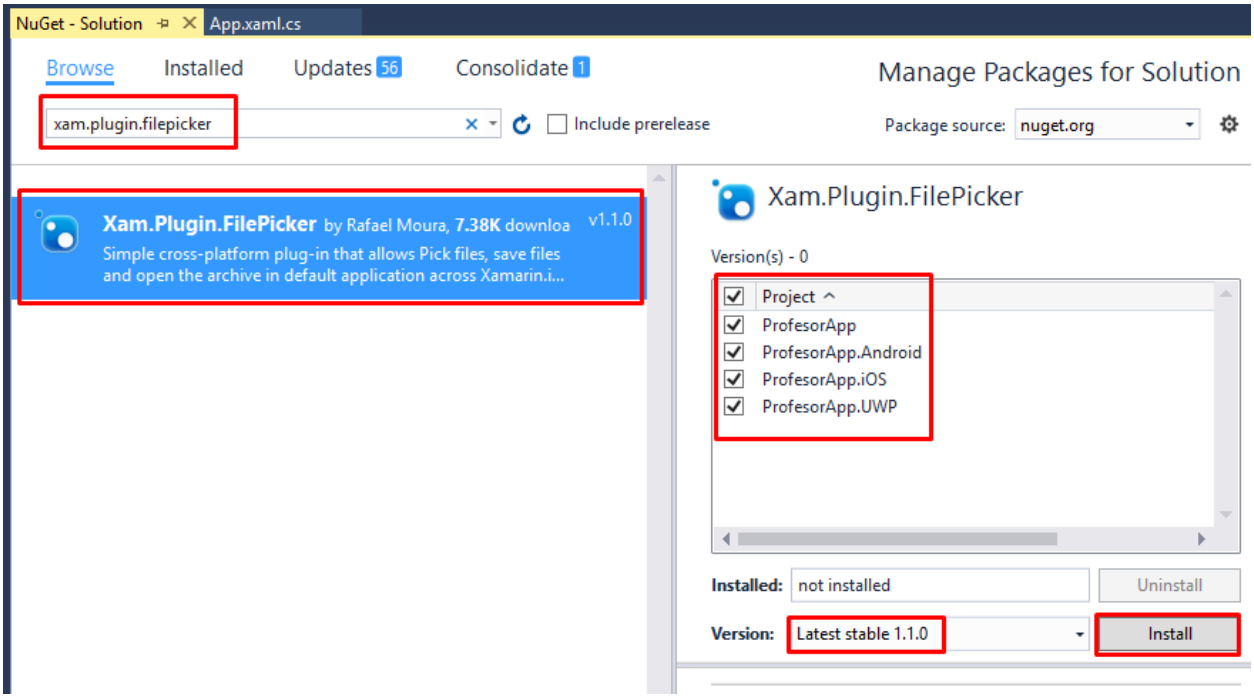
c) Microsoft.Net.Http



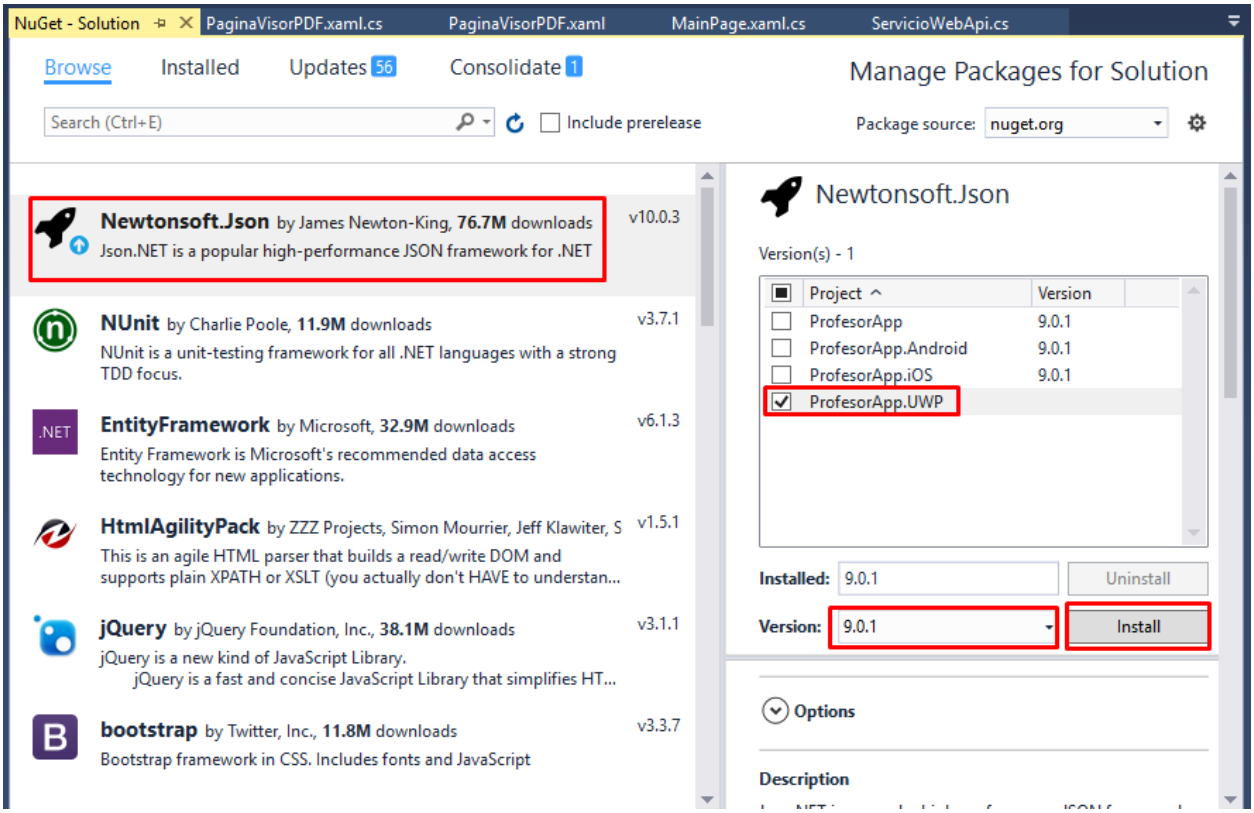
d) WindowsAzure.Storage



e) Xam.Plugin.FilePicker



f) **Newtonsoft.Json (actualizar el paquete de UWP)**



Paso 8. En la carpeta Clases, agrega las siguientes clases:

a) **Alumno**

```
namespace ProfesorApp.Clases
{
    public class Alumno
    {
        public int Id { get; set; }
        public string Nombre { get; set; }
        public string FotoURL { get; set; }
        public string Usuario { get; set; }
        public string Password { get; set; }
        public string FotoURLSAS { get; set; }
    }
}
```

b) Tarea

```
using System;

namespace ProfesorApp.Clases
{
    public class Tarea
    {
        public int Id { get; set; }
        public string Titulo { get; set; }
        public string ArchivoURL { get; set; }
        public DateTime FechaPublicacion { get; set; } = DateTime.Now;
        public DateTime FechaLimite { get; set; } = DateTime.Now;

        public string FechaPublicacionDate { get { return
FechaPublicacion.ToString("dd/MM/yyyy"); } }
        public string FechaLimiteDate { get { return FechaLimite.ToString("dd/MM/yyyy");
} }
    }
}
```

c) TareaAlumno

```
using System;

namespace ProfesorApp.Clases
{
    public class TareaAlumno
    {
        public int IdTarea { get; set; }
        public int IdAlumno { get; set; }

        public string Mensaje { get; set; }
        public string ArchivoURL { get; set; }
        public DateTime Fecha { get; set; }
        public int Calificacion { get; set; }
        public bool Evaluado { get; set; }

        public Tarea Tarea { get; set; }
        public Alumno Alumno { get; set; }

        public string FechaRespuestaDate { get { return Fecha.ToString("dd/MM/yyyy"); } }
        public string EvaluadoString { get { return Evaluado ? "Tarea evaluada" :
"Pendiente de evaluar"; } }
        public string MensajeCorto { get { return (Mensaje.Length > 50) ?
Mensaje.Substring(0, 50) : Mensaje; } }
    }
}
```

Paso 9. En la carpeta **Servicios**, agrega las siguientes clases:

a) **ServicioStorage**

```
using Microsoft.WindowsAzure.Storage.Blob;
using System.Threading.Tasks;
using System.IO;
using System;

namespace ProfesorApp.Servicios
{
    public class ServicioStorage
    {
        const string StorageURL = "este valor lo debes establecer";
        const string ContainerAlumno = "alumnos";
        const string ContainerTarea = "tareas-asignadas";
        const string ContainerTareaAlumno = "tareas-alumnos";
        const string SASQueryString = "este valor lo debes establecer";

        public async Task<string> UploadTarea(int id, Stream stream)
        {
            string blobSAS = $"{StorageURL}/{ContainerTarea}/{id}.pdf{SASQueryString}";
            return await UploadBlob(blobSAS, stream);
        }

        public async Task<string> UploadAlumno(int id, Stream stream)
        {
            string blobSAS = $"{StorageURL}/{ContainerAlumno}/{id}.jpg{SASQueryString}";
            return await UploadBlob(blobSAS, stream);
        }

        public async Task<Stream> DownloadAlumno(int id)
        {
            string blobSAS = $"{StorageURL}/{ContainerAlumno}/{id}.jpg{SASQueryString}";
            return await DownloadBlob(blobSAS);
        }

        public string GetFullDownloadTareaURL(int id)
        {
            return $"{StorageURL}/{ContainerTarea}/{id}.pdf{SASQueryString}";
        }

        public string GetFullDownloadAlumnoURL(int id)
        {
            return $"{StorageURL}/{ContainerAlumno}/{id}.pdf{SASQueryString}";
        }

        public string GetFullDownloadTareaAlumnoURL(int idTarea, int idAlumno)
        {
            return
$"{StorageURL}/{ContainerTareaAlumno}/{idTarea}_{idAlumno}.pdf{SASQueryString}";
        }
```

```
        public async Task<Stream> DownloadTarea(int id)
        {
            string blobSAS = $"{StorageURL}/{ContainerTarea}/{id}.pdf{SASQueryString}";
            return await DownloadBlob(blobSAS);
        }

        public async Task<Stream> DownloadTareaAlumnos(int idTarea, int idAlumno)
        {
            string blobSAS =
$"{StorageURL}/{ContainerTareaAlumno}/{idTarea}_{idAlumno}.pdf{SASQueryString}";
            return await DownloadBlob(blobSAS);
        }

        private async Task<Stream> DownloadBlob(string blobSAS)
        {
            try
            {
                CloudBlockBlob blob = new CloudBlockBlob(new Uri(blobSAS));
                MemoryStream stream = new MemoryStream();
                await blob.DownloadToStreamAsync(stream);
                return stream;
            }
            catch (Exception exc)
            {
                string msgError = exc.Message;
                return null;
            }
        }

        private async Task<string> UploadBlob(string blobSAS, Stream stream)
        {
            string url = "";

            try
            {
                CloudBlockBlob blob = new CloudBlockBlob(new Uri(blobSAS));

                using (stream)
                {
                    await blob.UploadFromStreamAsync(stream);
                    url = blob.StorageUri.PrimaryUri.AbsoluteUri;
                }
            }
            catch (Exception exc)
            {
                string msgError = exc.Message;
            }

            return url;
        }
    }
}
```

b) ServicioFilePicker

```
using Plugin.FilePicker;
using System;
using System.IO;
using System.Threading.Tasks;

namespace ProfesorApp.Servicios
{
    public class ServicioFilePicker
    {
        public async Task<MemoryStream> GetFile()
        {
            try
            {
                var data = await CrossFilePicker.Current.PickFile();
                var bytes = data.DataArray;
                return new MemoryStream(bytes);
            }
            catch(Exception ex)
            {
                return null;
            }
        }
    }
}
```

c) ServicioWebApi

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Threading.Tasks;
using Newtonsoft.Json;
using System.Net.Http;
using ProfesorApp.Clases;
using System.Net.Http.Headers;
using System.Net;


namespace ProfesorApp.Servicios
{
    public class ServicioWebApi
    {
        const string WebApiURL = "este valor lo debes establecer";

        private static HttpClient Cliente = new HttpClient();

        public async Task<List<Alumno>> GetAlumnos()
        {
            List<Alumno> datos = null;
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));
```

```
            var url = $"{WebApiURL}/api/Alumnos/";
            var respuesta = await Cliente.GetAsync(url);

            if (respuesta.StatusCode == HttpStatusCode.OK)
            {
                var json = await respuesta.Content.ReadAsStringAsync();
                datos = JsonConvert.DeserializeObject<List<Alumno>>(json);

                var servicioStorage = new ServicioStorage();

                foreach (var alumno in datos)
                {
                    alumno.FotoURLSAS =
servicioStorage.GetFullDownloadAlumnoURL(alumno.Id);
                }
            }

            return datos;
        }

        public async Task<Alumno> GetAlumno(int id)
        {
            Alumno dato = null;
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"{WebApiURL}/api/Alumnos/{id}";
            var respuesta = await Cliente.GetAsync(url);

            if (respuesta.StatusCode == HttpStatusCode.OK)
            {
                var json = await respuesta.Content.ReadAsStringAsync();
                dato = JsonConvert.DeserializeObject<Alumno>(json);
            }

            return dato;
        }

        public async Task<Alumno> AddAlumno(Alumno info)
        {
            Alumno dato = null;
            Cliente.BaseAddress = new Uri(WebApiURL);
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"/api/Alumnos/";
            var jsonContent = JsonConvert.SerializeObject(info);
            var respuesta = await Cliente.PostAsync(url, new
StringContent(jsonContent.ToString(), Encoding.UTF8, "application/json"));

            //if (respuesta.StatusCode == HttpStatusCode.Created)
            {
                var json = await respuesta.Content.ReadAsStringAsync();
                dato = JsonConvert.DeserializeObject<Alumno>(json);
            }
```

```
            return dato;
        }

        public async Task<Alumno> UpdateAlumno(Alumno info)
        {
            Alumno dato = null;
            Cliente.BaseAddress = new Uri(WebApiURL);
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"/api/Alumnos/{info.Id}";
            var jsonContent = JsonConvert.SerializeObject(info);
            var respuesta = await Cliente.PutAsync(url, new
StringContent(jsonContent.ToString(), Encoding.UTF8, "application/json"));

            //if (respuesta.StatusCode == HttpStatusCode.Created)
            {
                var json = await respuesta.Content.ReadAsStringAsync();
                dato = JsonConvert.DeserializeObject<Alumno>(json);
            }

            return dato;
        }

        public async Task<bool> DeleteAlumno(int id)
        {
            Cliente.BaseAddress = new Uri(WebApiURL);
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"/api/Alumnos/{id}";
            var respuesta = await Cliente.DeleteAsync(url);
            return respuesta.IsSuccessStatusCode;
        }

        public async Task<List<Tarea>> GetTareas()
        {
            List<Tarea> datos = null;
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"{WebApiURL}/api/Tareas/";
            var respuesta = await Cliente.GetAsync(url);

            if (respuesta.StatusCode == HttpStatusCode.OK)
            {
                var json = await respuesta.Content.ReadAsStringAsync();
                datos = JsonConvert.DeserializeObject<List<Tarea>>(json);
            }

            return datos;
        }

        public async Task<Tarea> GetTarea(int id)
        {
```

```
            Tarea dato = null;
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"{WebApiURL}/api/Tareas/{id}";
            var respuesta = await Cliente.GetAsync(url);

            if (respuesta.StatusCode == HttpStatusCode.OK)
            {
                var json = await respuesta.Content.ReadAsStringAsync();
                dato = JsonConvert.DeserializeObject<Tarea>(json);
            }

            return dato;
        }

        public async Task<Tarea> AddTarea(Tarea info)
        {
            Tarea dato = null;
            Cliente.BaseAddress = new Uri(WebApiURL);
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"/api/Tareas/";
            var jsonContent = JsonConvert.SerializeObject(info);
            var respuesta = await Cliente.PostAsync(url, new
StringContent(jsonContent.ToString(), Encoding.UTF8, "application/json"));

            //if (respuesta.StatusCode == HttpStatusCode.Created)
            {
                var json = await respuesta.Content.ReadAsStringAsync();
                dato = JsonConvert.DeserializeObject<Tarea>(json);
            }

            return dato;
        }

        public async Task<Tarea> UpdateTarea(Tarea info)
        {
            Tarea dato = null;
            Cliente.BaseAddress = new Uri(WebApiURL);
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"/api/Tareas/{info.Id}";
            var jsonContent = JsonConvert.SerializeObject(info);
            var respuesta = await Cliente.PutAsync(url, new
StringContent(jsonContent.ToString(), Encoding.UTF8, "application/json"));

            //if (respuesta.StatusCode == HttpStatusCode.Created)
            {
                var json = await respuesta.Content.ReadAsStringAsync();
                dato = JsonConvert.DeserializeObject<Tarea>(json);
            }
```

```
            return dato;
        }

        public async Task<bool> DeleteTarea(int id)
        {
            Cliente.BaseAddress = new Uri(WebApiURL);
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"/api/Tareas/{id}";
            var respuesta = await Cliente.DeleteAsync(url);
            return respuesta.IsSuccessStatusCode;
        }

        public async Task<List<TareaAlumno>> GetTareaAlumnos()
        {
            List<TareaAlumno> datos = null;
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"{WebApiURL}/api/TareaAlumnos/";
            var respuesta = await Cliente.GetAsync(url);

            if (respuesta.StatusCode == HttpStatusCode.OK)
            {
                var json = await respuesta.Content.ReadAsStringAsync();
                datos = JsonConvert.DeserializeObject<List<TareaAlumno>>(json);
            }

            return datos;
        }

        public async Task<TareaAlumno> GetTareaAlumno(int idTarea, int idAlumno)
        {
            TareaAlumno dato = null;
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"{WebApiURL}/api/TareaAlumnos/{idTarea}/{idAlumno}";
            var respuesta = await Cliente.GetAsync(url);

            if (respuesta.StatusCode == HttpStatusCode.OK)
            {
                var json = await respuesta.Content.ReadAsStringAsync();
                dato = JsonConvert.DeserializeObject<TareaAlumno>(json);
            }

            return dato;
        }

        public async Task<TareaAlumno> AddTareaAlumno(TareaAlumno info)
        {
            TareaAlumno dato = null;
            Cliente.BaseAddress = new Uri(WebApiURL);
            Cliente.DefaultRequestHeaders.Accept.Clear();
```

```
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"/api/TareaAlumnos/";
            var jsonContent = JsonConvert.SerializeObject(info);
            var respuesta = await Cliente.PostAsync(url, new
StringContent(jsonContent.ToString(), Encoding.UTF8, "application/json"));

            //if (respuesta.StatusCode == HttpStatusCode.Created)
            {
                var json = await respuesta.Content.ReadAsStringAsync();
                dato = JsonConvert.DeserializeObject<TareaAlumno>(json);
            }

            return dato;
        }

        public async Task<TareaAlumno> UpdateTareaAlumno(TareaAlumno info)
        {
            TareaAlumno dato = null;
            Cliente.BaseAddress = new Uri(WebApiURL);
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"/api/TareaAlumnos/{info.IdTarea}/{info.IdAlumno}";
            var jsonContent = JsonConvert.SerializeObject(info);
            var respuesta = await Cliente.PutAsync(url, new
StringContent(jsonContent.ToString(), Encoding.UTF8, "application/json"));

            //if (respuesta.StatusCode == HttpStatusCode.Created)
            {
                var json = await respuesta.Content.ReadAsStringAsync();
                dato = JsonConvert.DeserializeObject<TareaAlumno>(json);
            }

            return dato;
        }

        public async Task<bool> DeleteTareaAlumno(int idTarea, int idAlumno)
        {
            Cliente.BaseAddress = new Uri(WebApiURL);
            Cliente.DefaultRequestHeaders.Accept.Clear();
            Cliente.DefaultRequestHeaders.Accept.Add(new
MediaTypeWithQualityHeaderValue("application/json"));

            var url = $"/api/TareaAlumnos/{idTarea}/{idAlumno}";
            var respuesta = await Cliente.DeleteAsync(url);
            return respuesta.IsSuccessStatusCode;
        }
    }
}
```

Paso 10. En la carpeta **Paginas**, agrega las siguientes páginas:

a) PáginaMenu

Código XAML

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="ProfesorApp.Paginas.PaginaMenu">
    <ContentPage.ToolbarItems>
        <ToolbarItem x:Name="Tareas" Text="Tareas" Order="Primary" Priority="0"
Clicked="Tareas_Clicked"/>
        <ToolbarItem x:Name="Alumnos" Text="Alumnos" Order="Primary" Priority="2"
Clicked="Alumnos_Clicked"/>
        <ToolbarItem x:Name="Respuestas" Text="Respuestas" Order="Primary" Priority="2"
Clicked="Respuestas_Clicked"/>
    </ContentPage.ToolbarItems>
</ContentPage>
```

Código C#:

```
using System;

using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace ProfesorApp.Paginas
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class PaginaMenu : ContentPage
    {
        public PaginaMenu()
        {
            InitializeComponent();
        }

        private async void Tareas_Clicked(object sender, EventArgs e)
        {
            await Navigation.PushAsync(new PaginaListaTareas());
        }

        private async void Alumnos_Clicked(object sender, EventArgs e)
        {
            await Navigation.PushAsync(new PaginaListaAlumnos());
        }

        private async void Respuestas_Clicked(object sender, EventArgs e)
        {
            await Navigation.PushAsync(new PaginaListaTareasAlumnos());
        }
    }
}
```

b) PaginaListaTareas

Código XAML

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="ProfesorApp.Paginas.PaginaListaTareas">
    <ScrollView>
        <StackLayout>
            <ActivityIndicator x:Name="activityIndicator" Color="Blue"/>
            <ListView x:Name="lsvTareas" ItemSelected="lsvTareas_ItemSelected">
                <ListView.ItemTemplate>
                    <DataTemplate>
                        <TextCell Text="{Binding Titulo}" TextColor="Blue" />
                    </DataTemplate>
                </ListView.ItemTemplate>
            </ListView>
        </StackLayout>
    </ScrollView>
    <ContentPage.ToolbarItems>
        <ToolbarItem x:Name="Agregar" Text="Agregar" Order="Primary" Priority="0"
Clicked="Agregar_Clicked"/>
    </ContentPage.ToolbarItems>
</ContentPage>
```

Código C#:

```
using System;
using ProfesorApp.Servicios;
using ProfesorApp.Clases;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace ProfesorApp.Paginas
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class PaginaListaTareas : ContentPage
    {
        public PaginaListaTareas()
        {
            InitializeComponent();
        }

        private void ActualizarActivityIndicator(bool estado)
        {
            activityIndicator.IsRunning = estado;
            activityIndicator.IsEnabled = estado;
            activityIndicator.IsVisible = estado;
        }

        protected async override void OnAppearing()
        {
            base.OnAppearing();

            ActualizarActivityIndicator(true);
```

```
            var servicioWebApi = new ServicioWebApi();
            lsvTareas.ItemsSource = await servicioWebApi.GetTareas();

            ActualizarActivityIndicator(false);
        }

        private async void lsvTareas_ItemSelected(object sender,
SelectedItemChangedEventArgs e)
        {
            try
            {
                Tarea dato = (Tarea)e.SelectedItem;
                await Navigation.PushAsync(new PaginaDetalleTarea(dato));
            }
            catch (Exception ex)
            {
            }
        }

        private async void Agregar_Clicked(object sender, EventArgs e)
        {
            await Navigation.PushAsync(new PaginaDetalleTarea(new Tarea()));
        }
    }
}
```

c) PaginaDetalleTarea

Código XAML

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="ProfesorApp.Paginas.PaginaDetalleTarea">
    <StackLayout Padding="10" Spacing="10" BackgroundColor="White">
        <Label Text="Titulo:" FontSize="Medium" TextColor="Black"
HorizontalOptions="Start" Margin="10" HorizontalTextAlignment="Start"/>
        <Entry Text="{Binding Titulo}" HorizontalOptions="FillAndExpand" FontSize="20"
TextColor="White" BackgroundColor="Black" Margin="10,0" HorizontalTextAlignment="Start"
FontAttributes="Bold"/>

        <Label Text="Fecha límite:" FontSize="Medium" TextColor="Black"
HorizontalOptions="Start" Margin="10" HorizontalTextAlignment="Start"/>
        <DatePicker Date="{Binding FechaLimite}" Format="dd/MM/yyyy"
BackgroundColor="White" TextColor="Black" Margin="10,0"/>

        <Label Text="Fecha publicación:" FontSize="Medium" TextColor="Black"
HorizontalOptions="Start" Margin="10" HorizontalTextAlignment="Start"/>
        <Label Text="{Binding FechaPublicacionDate}" FontSize="20" TextColor="Black"
HorizontalOptions="Start" Margin="12,0" HorizontalTextAlignment="Start"/>

        <Button x:Name="btnArchivo" Text="Seleccionar archivo" HorizontalOptions="Center"
TextColor="Black" BackgroundColor="White" Clicked="btnArchivo_Clicked"
FontAttributes="Bold" FontSize="20" BorderColor="Black" BorderWidth="10"/>

        <ActivityIndicator x:Name="activityIndicator" Color="Blue" Margin="10"/>
    </StackLayout>

    <ContentPage.ToolbarItems>
        <ToolbarItem x:Name="Guardar" Text="Guardar" Order="Primary" Priority="0"
Clicked="Guardar_Clicked"/>
        <ToolbarItem x:Name="Eliminar" Text="Eliminar" Order="Primary" Priority="1"
Clicked="Eliminar_Clicked"/>
        <ToolbarItem x:Name="Ver" Text="Archivo" Order="Primary" Priority="2"
Clicked="Ver_Clicked"/>
    </ContentPage.ToolbarItems>
</ContentPage>
```

Código C#:

```
using System;
using ProfesorApp.Servicios;
using ProfesorApp.Clases;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;
using System.IO;

namespace ProfesorApp.Paginas
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class PaginaDetalleTarea : ContentPage
    {
```

```
        Tarea dato;
        MemoryStream stream;

        public PaginaDetalleTarea(Tarea dato)
        {
            InitializeComponent();

            ActualizarActivityIndicator(true);

            this.dato = dato;
            this.BindingContext = dato;

            ActualizarActivityIndicator(false);
        }

        private void ActualizarActivityIndicator(bool estado)
        {
            activityIndicator.IsRunning = estado;
            activityIndicator.IsEnabled = estado;
            activityIndicator.IsVisible = estado;
        }

        private async void btnArchivo_Clicked(object sender, EventArgs e)
        {
            ServicioFilePicker servicioFilePicker = new ServicioFilePicker();
            stream = await servicioFilePicker.GetFile();
        }

        private async void Guardar_Clicked(object sender, EventArgs e)
        {
            ActualizarActivityIndicator(true);

            var servicioStorage = new ServicioStorage();
            var servicioWebApi = new ServicioWebApi();

            if (dato.Id == 0)
            {
                dato = await servicioWebApi.AddTarea(dato);
            }

            dato.ArchivoURL = await servicioStorage.UploadTarea(dato.Id, stream);
            await servicioWebApi.UpdateTarea(dato);

            ActualizarActivityIndicator(false);

            await DisplayAlert("Información", "Dato registrado con éxito", "OK");
            await Navigation.PopAsync();
        }

        private async void Eliminar_Clicked(object sender, EventArgs e)
        {
            if (dato.Id > 0)
            {
                if (await DisplayAlert("Eliminar", "¿Deseas eliminar el registro?", "Si",
"No"))
                {
                    ActualizarActivityIndicator(true);
```

```
                    var servicioWebApi = new ServicioWebApi();
                    await servicioWebApi.DeleteTarea(dato.Id);

                    ActualizarActivityIndicator(false);

                    await DisplayAlert("Información", "Dato eliminado con éxito", "OK");
                    await Navigation.PopAsync();
                }
            }
        }

        private void Ver_Clicked(object sender, EventArgs e)
        {
            if (dato.Id > 0)
            {
                var servicioStorage = new ServicioStorage();
                Device.OpenUri(new
Uri(servicioStorage.GetFullDownloadTareaURL(dato.Id)));
                //var stream = await servicioStorage.DownloadTarea(dato.Id);
            }
        }
    }
}
```

d) PaginaListaAlumnos

Código XAML

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="ProfesorApp.Paginas.PaginaListaAlumnos">
    <ScrollView>
        <StackLayout>
            <ActivityIndicator x:Name="activityIndicator" Color="Blue"/>
            <ListView x:Name="lsvAlumnos" ItemSelected="lsvAlumnos_ItemSelected">
                <ListView.ItemTemplate>
                    <DataTemplate>
                        <ImageCell ImageSource="{Binding FotoURLSAS}" Text="{Binding
Nombre}" TextColor="Blue"/>
                    </DataTemplate>
                </ListView.ItemTemplate>
            </ListView>
        </StackLayout>
    </ScrollView>
    <ContentPage.ToolbarItems>
        <ToolbarItem x:Name="Agregar" Text="Agregar" Order="Primary" Priority="0"
Clicked="Agregar_Clicked"/>
    </ContentPage.ToolbarItems>
</ContentPage>
```

Código C#:

```
using System;
using ProfesorApp.Servicios;
using ProfesorApp.Clases;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace ProfesorApp.Paginas
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class PaginaListaAlumnos : ContentPage
    {
        public PaginaListaAlumnos()
        {
            InitializeComponent();
        }

        private void ActualizarActivityIndicator(bool estado)
        {
            activityIndicator.IsRunning = estado;
            activityIndicator.IsEnabled = estado;
            activityIndicator.IsVisible = estado;
        }

        protected async override void OnAppearing()
        {
            base.OnAppearing();
```

```
            ActualizarActivityIndicator(true);

            var servicioWebApi = new ServicioWebApi();
            lsvAlumnos.ItemsSource = await servicioWebApi.GetAlumnos();

            ActualizarActivityIndicator(false);
        }

        private async void lsvAlumnos_ItemSelected(object sender,
SelectedItemChangedEventArgs e)
        {
            try
            {
                Alumno dato = (Alumno)e.SelectedItem;
                await Navigation.PushAsync(new PaginaDetalleAlumno(dato));
            }
            catch (Exception ex)
            {
            }
        }

        private async void Agregar_Clicked(object sender, EventArgs e)
        {
            await Navigation.PushAsync(new PaginaDetalleAlumno(new Alumno()));
        }
    }
}
```

e) PaginaDetalleAlumno

Código XAML

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="ProfesorApp.Paginas.PaginaDetalleAlumno">
    <StackLayout Padding="10" Spacing="10" BackgroundColor="White">
        <Image Source="{Binding FotoURLSAS}" WidthRequest="200" HeightRequest="200"
Aspect="AspectFit" HorizontalOptions="Center" />

        <Label Text="Nombre:" FontSize="Medium" TextColor="Black"
HorizontalOptions="Start" Margin="10" HorizontalTextAlignment="Start"/>
        <Entry Text="{Binding Nombre}" HorizontalOptions="FillAndExpand" FontSize="20"
TextColor="White" BackgroundColor="Black" Margin="10,0" HorizontalTextAlignment="Start"
FontAttributes="Bold"/>

        <Label Text="Usuario:" FontSize="Medium" TextColor="Black"
HorizontalOptions="Start" Margin="10" HorizontalTextAlignment="Start"/>
        <Entry Text="{Binding Usuario}" HorizontalOptions="FillAndExpand" FontSize="20"
TextColor="White" BackgroundColor="Black" Margin="10,0" HorizontalTextAlignment="Start"
FontAttributes="Bold"/>

        <Label Text="Password:" FontSize="Medium" TextColor="Black"
HorizontalOptions="Start" Margin="10" HorizontalTextAlignment="Start"/>
        <Entry Text="{Binding Password}" IsPassword="True"
HorizontalOptions="FillAndExpand" FontSize="20" TextColor="White" BackgroundColor="Black"
Margin="10,0" HorizontalTextAlignment="Start" FontAttributes="Bold"/>

        <ActivityIndicator x:Name="activityIndicator" Color="Blue" Margin="10"/>
    </StackLayout>

    <ContentPage.ToolbarItems>
        <ToolbarItem x:Name="Guardar" Text="Guardar" Order="Primary" Priority="0"
Clicked="Guardar_Clicked"/>
        <ToolbarItem x:Name="Eliminar" Text="Eliminar" Order="Primary" Priority="1"
Clicked="Eliminar_Clicked"/>
    </ContentPage.ToolbarItems>
</ContentPage>
```

Código C#:

```
using System;
using ProfesorApp.Servicios;
using ProfesorApp.Clases;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace ProfesorApp.Paginas
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class PaginaDetalleAlumno : ContentPage
    {
        Alumno dato;
```

```
        public PaginaDetalleAlumno(Alumno dato)
        {
            InitializeComponent();

            ActualizarActivityIndicator(true);

            this.dato = dato;
            this.BindingContext = dato;

            ActualizarActivityIndicator(false);
        }

        private void ActualizarActivityIndicator(bool estado)
        {
            activityIndicator.IsRunning = estado;
            activityIndicator.IsEnabled = estado;
            activityIndicator.IsVisible = estado;
        }

        private async void Guardar_Clicked(object sender, EventArgs e)
        {
            ActualizarActivityIndicator(true);

            var servicioWebApi = new ServicioWebApi();

            if (dato.Id == 0)
                dato = await servicioWebApi.AddAlumno(dato);
            else
                await servicioWebApi.UpdateAlumno(dato);

            ActualizarActivityIndicator(false);

            await DisplayAlert("Información", "Dato registrado con éxito", "OK");
            await Navigation.PopAsync();
        }

        private async void Eliminar_Clicked(object sender, EventArgs e)
        {
            if (dato.Id > 0)
            {
                if (await DisplayAlert("Eliminar", "¿Deseas eliminar el registro?", "Si",
"No"))
                {
                    ActualizarActivityIndicator(true);

                    var servicioWebApi = new ServicioWebApi();
                    await servicioWebApi.DeleteAlumno(dato.Id);

                    ActualizarActivityIndicator(false);

                    await DisplayAlert("Información", "Dato eliminado con éxito", "OK");
                    await Navigation.PopAsync();
                }
            }
        }
    }
}
```

f) PaginaListaTareasAlumnos

Código XAML

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="ProfesorApp.Paginas.PaginaListaTareasAlumnos">
    <ScrollView>
        <StackLayout>
            <ActivityIndicator x:Name="activityIndicator" Color="Blue"/>
            <ListView x:Name="lsvTareasAlumnos"
ItemSelected="lsvTareasAlumnos_ItemSelected" HasUnevenRows="True">
                <ListView.ItemTemplate>
                    <DataTemplate>
                        <ViewCell>
                            <StackLayout BackgroundColor="White" Spacing="5" Padding="5">
                                <Label Text="{Binding MensajeCorto}"
LineBreakMode="WordWrap" FontSize="15" TextColor="#030303" HorizontalOptions="Start"
Margin="12,5,12,1" HorizontalTextAlignment="Start"/>
                                <Label Text="{Binding FechaRespuestaDate}"
LineBreakMode="WordWrap" FontSize="15" TextColor="#030303" HorizontalOptions="Start"
Margin="12,5,12,1" HorizontalTextAlignment="Start"/>
                                <StackLayout Orientation="Horizontal">
                                    <Label Text="{Binding Calificacion}"
LineBreakMode="WordWrap" FontSize="15" TextColor="#030303" HorizontalOptions="Start"
Margin="12,5,12,1" HorizontalTextAlignment="Start"/>
                                    <Label Text="{Binding EvaluadoString}"
LineBreakMode="WordWrap" FontSize="15" TextColor="#030303" HorizontalOptions="Start"
Margin="12,5,12,1" HorizontalTextAlignment="Start"/>
                                </StackLayout>
                            </StackLayout>
                        </ViewCell>
                    </DataTemplate>
                </ListView.ItemTemplate>
            </ListView>
        </StackLayout>
    </ScrollView>
</ContentPage>
```

Código C#:

```
using System;
using ProfesorApp.Servicios;
using ProfesorApp.Clases;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace ProfesorApp.Paginas
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class PaginaListaTareasAlumnos : ContentPage
    {
        public PaginaListaTareasAlumnos()
        {
            InitializeComponent();
```

```
        }

        private void ActualizarActivityIndicator(bool estado)
        {
            activityIndicator.IsRunning = estado;
            activityIndicator.IsEnabled = estado;
            activityIndicator.IsVisible = estado;
        }

        protected async override void OnAppearing()
        {
            base.OnAppearing();

            ActualizarActivityIndicator(true);

            var servicioWebApi = new ServicioWebApi();
            lsvTareasAlumnos.ItemsSource = await servicioWebApi.GetTareaAlumnos();

            ActualizarActivityIndicator(false);
        }

        private async void lsvTareasAlumnos_ItemSelected(object sender,
SelectedItemChangedEventArgs e)
        {
            try
            {
                TareaAlumno dato = (TareaAlumno)e.SelectedItem;
                await Navigation.PushAsync(new PaginaCalificarTareaAlumno(dato));
            }
            catch (Exception ex)
            {
            }
        }
    }
}
```

g) PaginaCalificarTareaAlumno

Código XAML

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="ProfesorApp.Paginas.PaginaCalificarTareaAlumno">

    <ScrollView>
        <StackLayout Padding="10" Spacing="10" BackgroundColor="White">
            <ActivityIndicator x:Name="activityIndicator" Color="Blue" Margin="10"/>

            <Label Text="{Binding Tarea.Titulo}" FontSize="20" TextColor="Black"
HorizontalOptions="Start" Margin="10" HorizontalTextAlignment="Start"
FontAttributes="Bold"/>
            <BoxView HeightRequest="1" BackgroundColor="Black"
HorizontalOptions="FillAndExpand"/>

            <StackLayout Orientation="Horizontal">
                <Image Source="{Binding FotoURLSAS}" WidthRequest="150"
HeightRequest="150" Aspect="AspectFit" HorizontalOptions="Center" />
                <Label Text="{Binding Alumno.Nombre}" FontSize="20" TextColor="Black"
HorizontalOptions="Start" Margin="12,0" VerticalOptions="Center"/>
            </StackLayout>
            <BoxView HeightRequest="1" BackgroundColor="Black"
HorizontalOptions="FillAndExpand"/>

            <Label Text="Mensaje del alumno:" FontSize="Medium" TextColor="Black"
HorizontalOptions="Start" Margin="10" HorizontalTextAlignment="Start"/>
            <Label Text="{Binding Mensaje}" FontSize="20" TextColor="Black"
HorizontalOptions="Start" Margin="12,0" HorizontalTextAlignment="Start"/>
            <BoxView HeightRequest="1" BackgroundColor="Black"
HorizontalOptions="FillAndExpand"/>

            <StackLayout Orientation="Horizontal">
                <Label Text="Fecha límite:" FontSize="Medium" TextColor="Black"
HorizontalOptions="Start" Margin="10" HorizontalTextAlignment="Start"/>
                <Label Text="{Binding Tarea.FechaLimiteDate}" FontSize="20"
TextColor="Black" HorizontalOptions="Start" Margin="12,0"
HorizontalTextAlignment="Start"/>
            </StackLayout>
            <BoxView HeightRequest="1" BackgroundColor="Black"
HorizontalOptions="FillAndExpand"/>

            <StackLayout Orientation="Horizontal">
                <Label Text="Fecha respuesta:" FontSize="Medium" TextColor="Black"
HorizontalOptions="Start" Margin="10" HorizontalTextAlignment="Start"/>
                <Label Text="{Binding FechaRespuestaDate}" FontSize="20"
TextColor="Black" HorizontalOptions="Start" Margin="12,0"
HorizontalTextAlignment="Start"/>
            </StackLayout>
            <BoxView HeightRequest="1" BackgroundColor="Black"
HorizontalOptions="FillAndExpand"/>

            <StackLayout Orientation="Horizontal" HorizontalOptions="FillAndExpand">
                <Label Text="Calificacion:" FontSize="Medium" TextColor="Black"
Margin="10" />
```

```
                <Label BindingContext="{x:Reference Name=sliderCalificacion}"
Text="{Binding Path=Value}" FontSize="20" TextColor="Black" HorizontalOptions="Start"
Margin="10" HorizontalTextAlignment="Start"/>
                <Slider x:Name="sliderCalificacion" Value="{Binding Calificacion}"
Minimum="0" Maximum="100" Margin="10" HorizontalOptions="FillAndExpand"/>
            </StackLayout>
            <BoxView HeightRequest="1" BackgroundColor="Black"
HorizontalOptions="FillAndExpand"/>
        </StackLayout>
    </ScrollView>

    <ContentPage.ToolbarItems>
        <ToolbarItem x:Name="Calificar" Text="Guardar" Order="Primary" Priority="0"
Clicked="Calificar_Clicked"/>
        <ToolbarItem x:Name="VerTarea" Text="Ver Tarea" Order="Primary" Priority="1"
Clicked="VerTarea_Clicked"/>
        <ToolbarItem x:Name="VerRespuesta" Text="Ver Respuesta" Order="Primary"
Priority="2" Clicked="VerRespuesta_Clicked"/>
    </ContentPage.ToolbarItems>
</ContentPage>
```

Código C#:

```
using System;
using ProfesorApp.Servicios;
using ProfesorApp.Clases;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace ProfesorApp.Paginas
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class PaginaCalificarTareaAlumno : ContentPage
    {
        TareaAlumno dato;

        public PaginaCalificarTareaAlumno(TareaAlumno dato)
        {
            InitializeComponent();

            this.dato = dato;
        }

        protected async override void OnAppearing()
        {
            base.OnAppearing();

            ActualizarActivityIndicator(true);

            ServicioWebApi servicioWebApi = new ServicioWebApi();
            dato = await servicioWebApi.GetTareaAlumno(dato.IdTarea, dato.IdAlumno);
            this.BindingContext = dato;

            ActualizarActivityIndicator(false);
        }
```

```
        private void ActualizarActivityIndicator(bool estado)
        {
            activityIndicator.IsRunning = estado;
            activityIndicator.IsEnabled = estado;
            activityIndicator.IsVisible = estado;
        }

        private async void Calificar_Clicked(object sender, EventArgs e)
        {
            ActualizarActivityIndicator(true);

            dato.Evaluado = true;

            var servicioWebApi = new ServicioWebApi();
            await servicioWebApi.UpdateTareaAlumno(dato);

            ActualizarActivityIndicator(false);

            await DisplayAlert("Información", "Dato registrado con éxito", "OK");
            await Navigation.PopAsync();
        }

        private void VerTarea_Clicked(object sender, EventArgs e)
        {
            var servicioStorage = new ServicioStorage();
            Device.OpenUri(new
Uri(servicioStorage.GetFullDownloadTareaURL(dato.IdTarea)));
        }

        private void VerRespuesta_Clicked(object sender, EventArgs e)
        {
            var servicioStorage = new ServicioStorage();
            Device.OpenUri(new
Uri(servicioStorage.GetFullDownloadTareaAlumnoURL(dato.IdTarea, dato.IdAlumno)));
        }
    }
}
```

Paso 11. Modifica App.xaml.cs:

```
        public App()
        {
            InitializeComponent();

            MainPage = new NavigationPage(new Paginas.PaginaMenu());
        }
```

Paso 12. Compila y ejecuta la app: