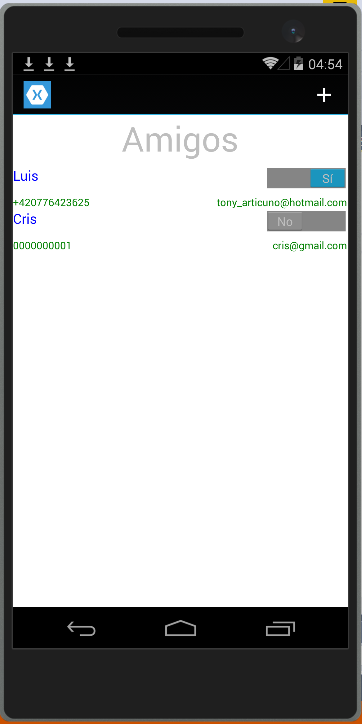
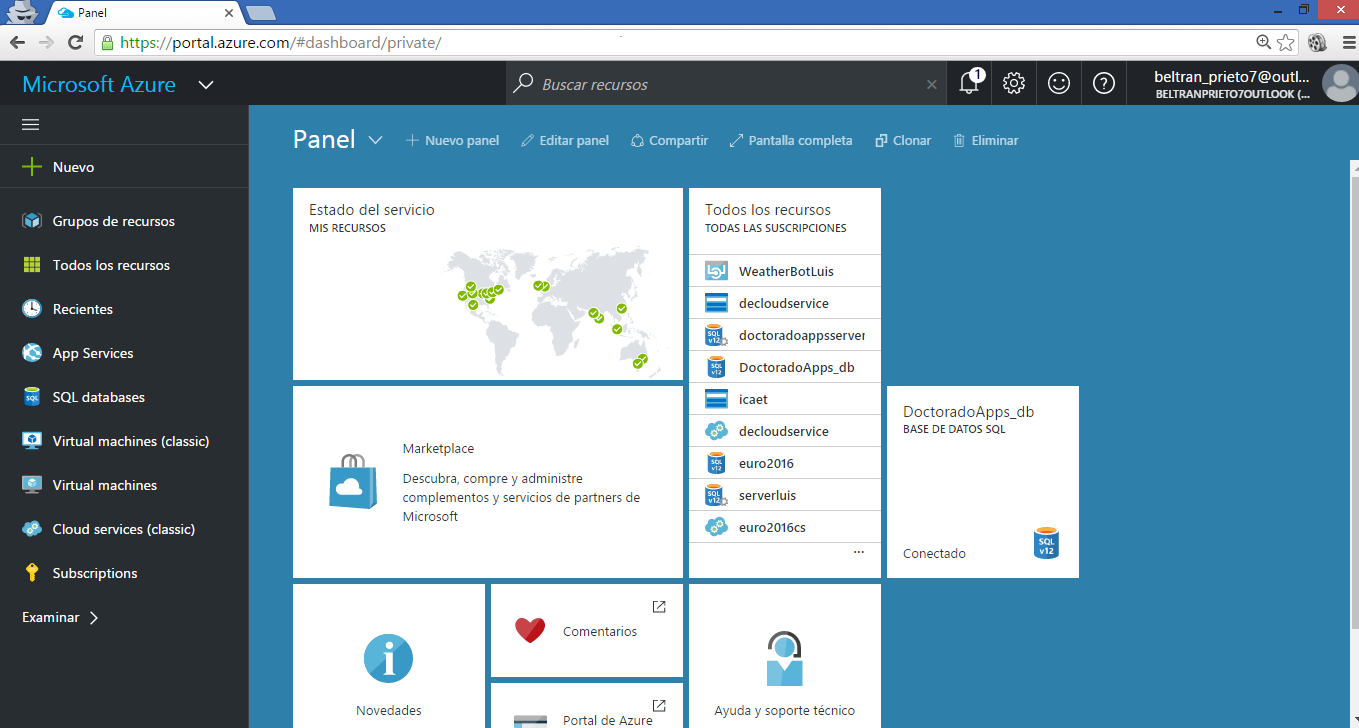
**Práctica 05 – Azure Mobile Services**

El objetivo de esta práctica es crear una aplicación que nos permita almacenar información de nuestros amigos en una base de datos en la nube creando un backend en Azure de una manera muy sencilla mediante Azure Mobile Services.

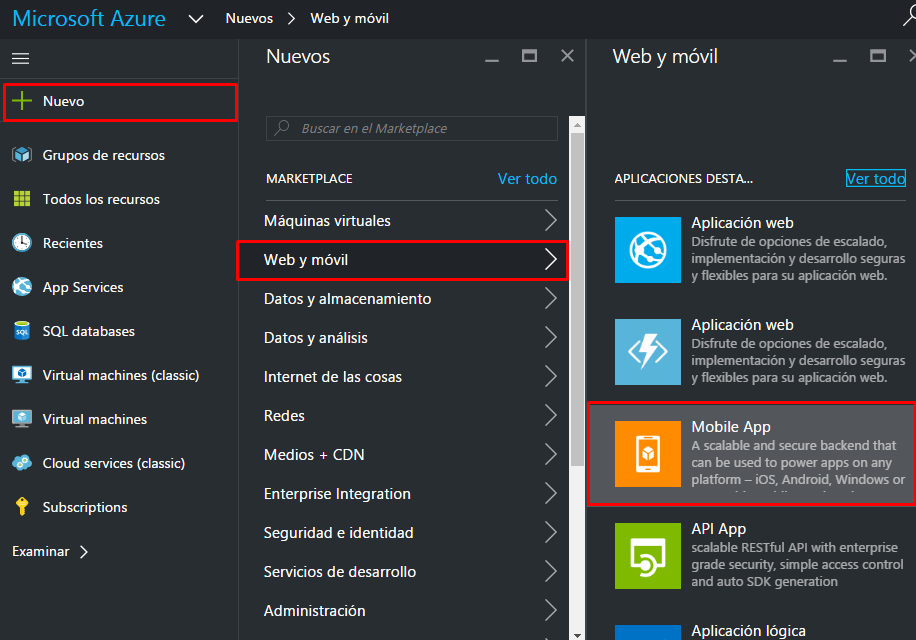
Para esta práctica necesitas una cuenta de Azure. Si no la tienes, ve al paso 21 para crear la app de Xamarin y utilizar los datos que ya he generado en esta práctica.

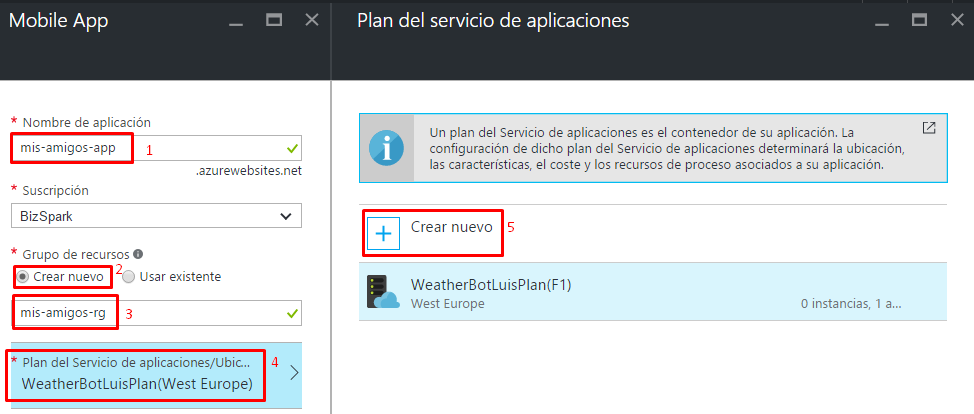
**Paso 1:** Ingresa al portal de Azure e inicia sesión (<http://portal.azure.com>).



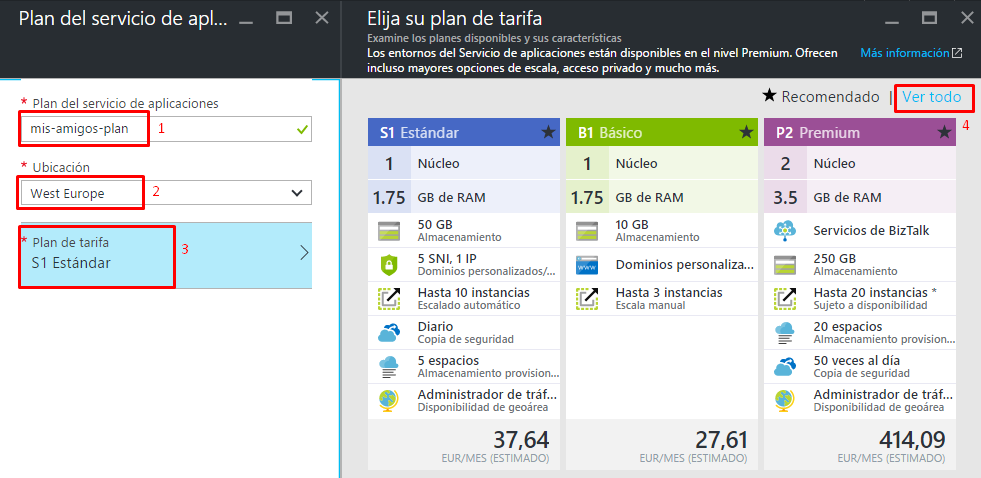
**Paso 2:** Utilizando el menúlateral, selecciona Nuevo – Web y móvil – Mobile App para crear un servicio móvil en Azure.



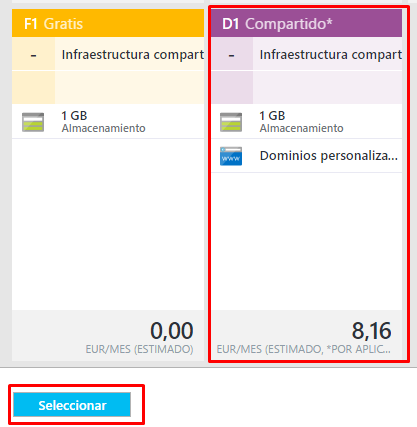
**Paso 3:** Ingresa **el nombre de la aplicación** (tiene que ser único, en mi caso he usado **mis-amigos-app**). Posteriormente, selecciona **Crear nuevo** en Grupo de recursos y escribe un nombre (**mis-amigos-rg**). Da clic en Plan del Servicio de aplicaciones y selecciona **Crear nuevo** en el panel que aparece a la derecha.



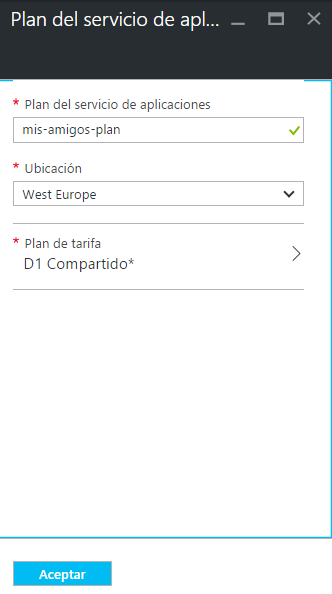
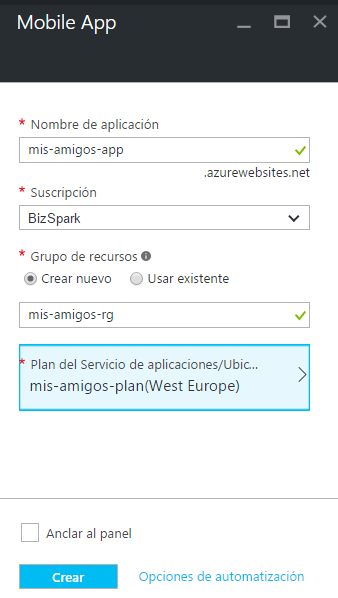
**Paso 4:** En la siguiente pantalla, ingresa **el nombre del plan del servicio** (por ejemplo, **mis-amigos-plan**), **una ubicación** cercana a donde estarán tus usuarios (en mi caso he seleccionado West Europe, pero puedes seleccionar Centro o sur de Estados Unidos si estás en México). Posteriormente, da clic en **Plan de Tarifa** y ahora da clic en **Ver todo**.



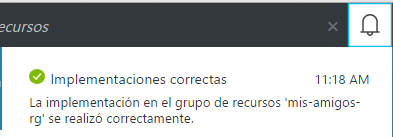
**Paso 5:** Puedes elegir el plan que gustes. En mi caso, dado que la app es sencilla, he seleccionado el plan **D1** **Compartido**, que es económico. También puedes seleccionar el gratuito si está disponible. Da clic en aceptar una vez seleccionado el plan de tu interés.



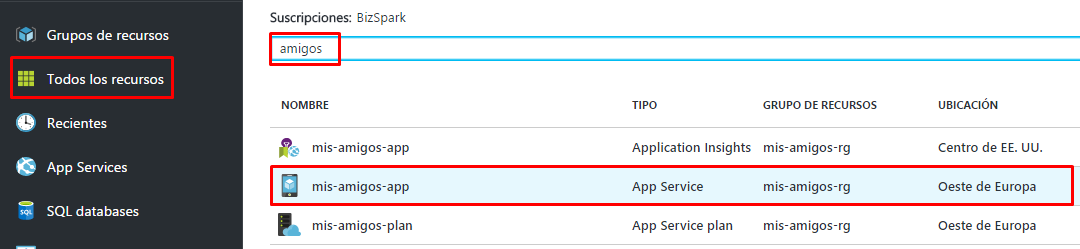
**Paso 6:** Una vez seleccionado el Plan, revisa que los datos ingresados sean correctos y da clic en **Aceptar** en cada pantalla. En mi caso, las categorías **Plan del servicio de aplicaciones** y **Mobile App** se muestran a continuación.

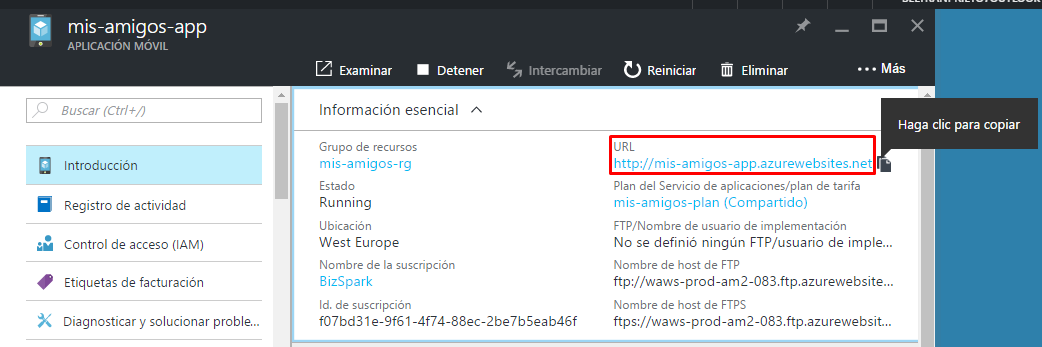
**Paso 7:** Ahora espera unos minutos en lo que se implementa el Mobile App. Recibirás una notificación cuando la implementación finalice.



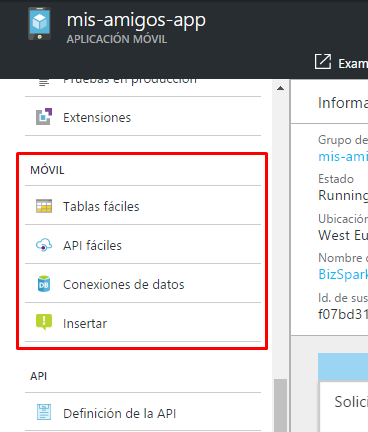
**Paso 8:** Ahora vamos a crear la base de datos y su conexión. Primero, da clic en la categoría **Todos los recursos** que aparece a la izquierda y posteriormente **busca** el App Service recién creado. Da clic en este recurso una vez identificado (revisa la columna **Tipo** para asegurar que estás seleccionando el recurso adecuado).



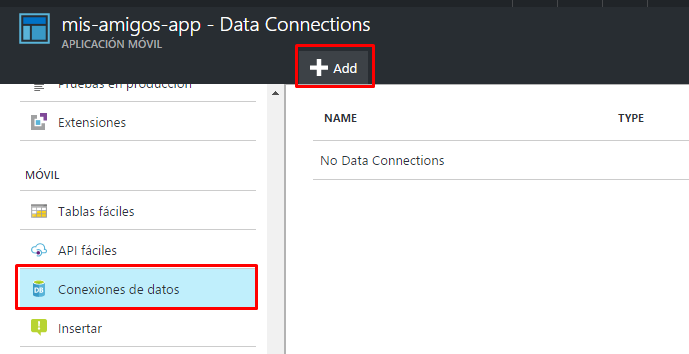
**Paso 9:** En la pantalla principal de la aplicación móvil seleccionada, identifica la **URL del servicio**, cópiala al portapapeles y pégala en algún lado. La utilizaremos posteriormente.



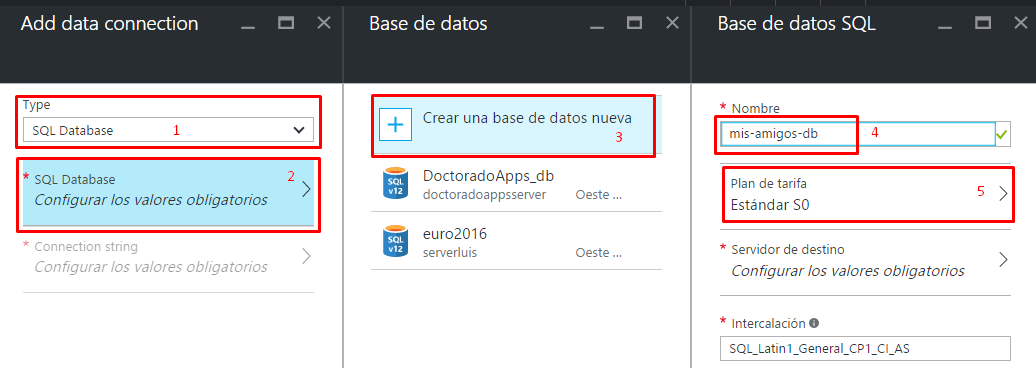
**Paso 10:** En el menú de la aplicación móvil (aparece a la izquierda), busca la categoría **Móvil**, la cual contiene 4 elementos: **Tablas fáciles**, **API fáciles**, **Conexiones de datos** e **Insertar**.



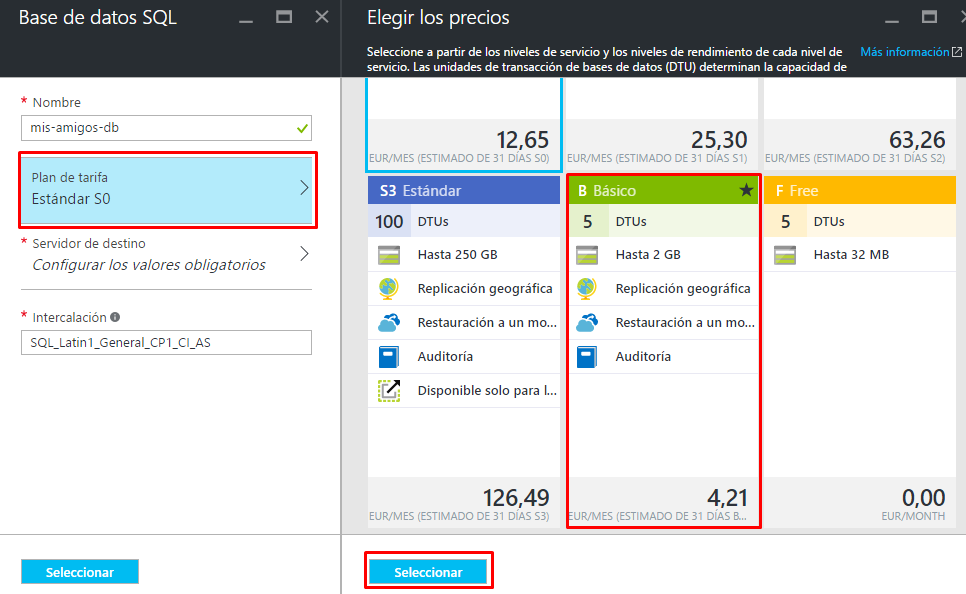
**Paso 11:** Selecciona **Conexiones de datos** y da clic en el botón **Add**.



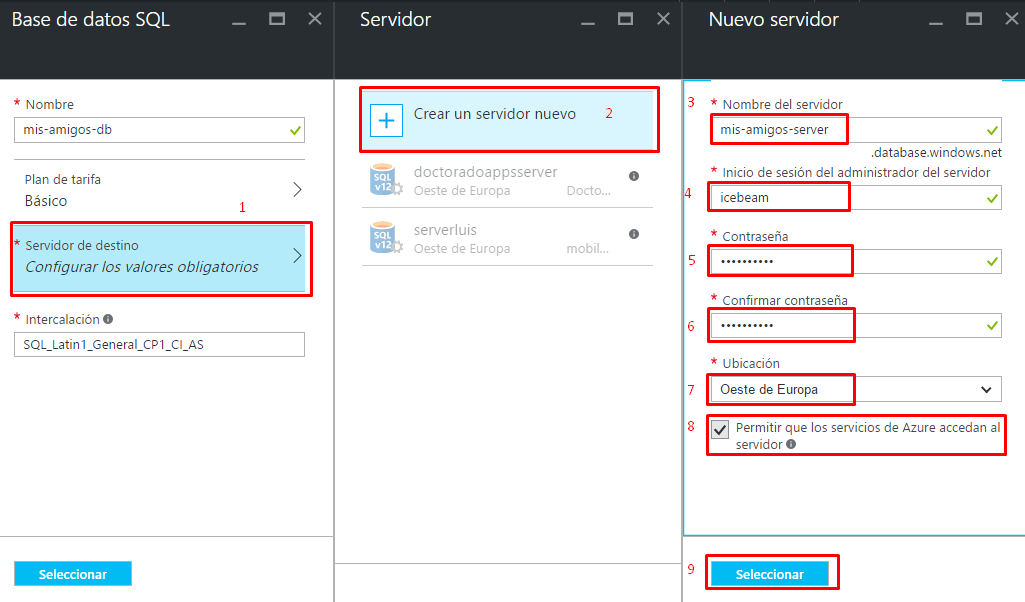
**Paso 12:** Selecciona **SQL Database** en el tipo de conexión; da clic en **Configurar los valores obligatorios**, y selecciona **Crear una base de datos nueva**.Ingresa un **nombre** parala base de datos (**mis-amigos-db**) y da clic en **Plan de tarifa**.



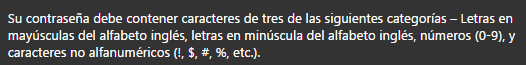
**Paso 13:** En el panel que aparece, selecciona un plan de tarifa que se ajuste a tus necesidades. Como la app es muy sencilla, el **Plan Básico** bastara, así que es el que he seleccionado. Da clic en **Seleccionar**.



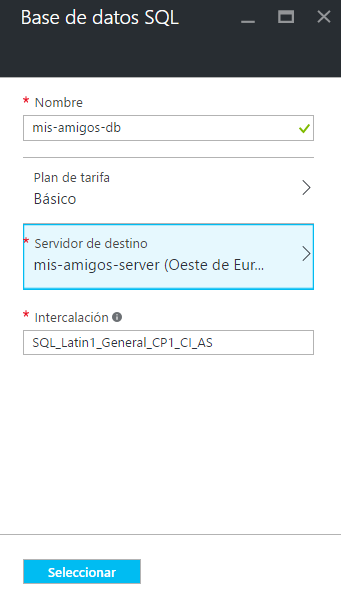
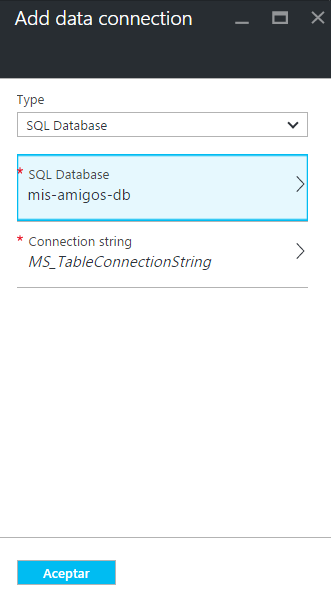
**Paso 14:** Ahora da clic en **Servidor de destino** y selecciona **Crear un servidor nuevo**. Ingresa la información que se solicita: **Nombre del servidor** (**mis-amigos-server**), **un usuario para iniciar sesión,** **la contraseña y su confirmación**, **la ubicación** del servidor (elige la misma ubicación que hayas seleccionado en el paso 4 para no incurrir en costos de transferencia entre servidores) y marca la casilla **Permitir que los servicios de Azure accedan al servidor** (con esto, estás otorgando al servicio móvil autorización de acceso al servidor). Finalmente, da clic en **Seleccionar**.



**NOTA: Revisa a continuación las reglas que debe cumplir tu contraseña para que sea válida.**

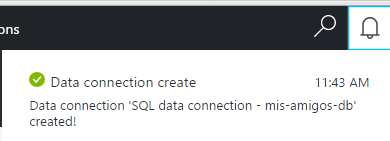


**Paso 15:** Una vez creado el servidor, revisa que la información que hayas ingresado sea la correcta. En caso afirmativo, da clic en **Seleccionar**. Lo mismo aplica para la pantalla **Add a data connection**.

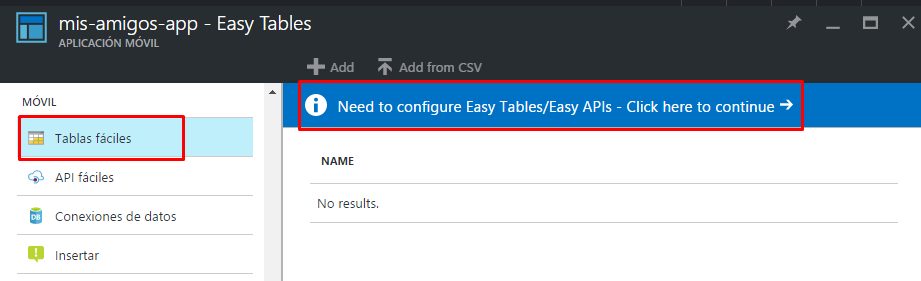
 

NOTA: En el caso de la cadena de conexión (**Connection string**), debemos mantener el nombre **MS\_TableConnectionString**, pues el servicio móvil necesita una cadena de conexión con este nombre a fin de que el backend funcione adecuadamente.

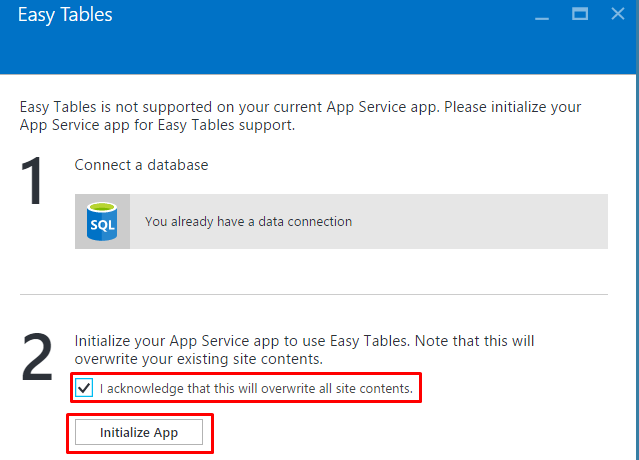
**Paso 16:** Espera por la confirmación de creación de la cadena de conexión. Recibirás una notificación cuando esto suceda.



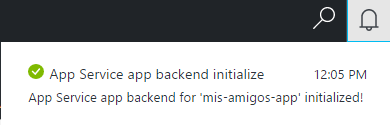
**Paso 17:** Accede de nuevo a la categoría Móvil. En esta ocasión, selecciona **Tablas fáciles** y da clic en el botón azul **Need to configure Easy Tables.**



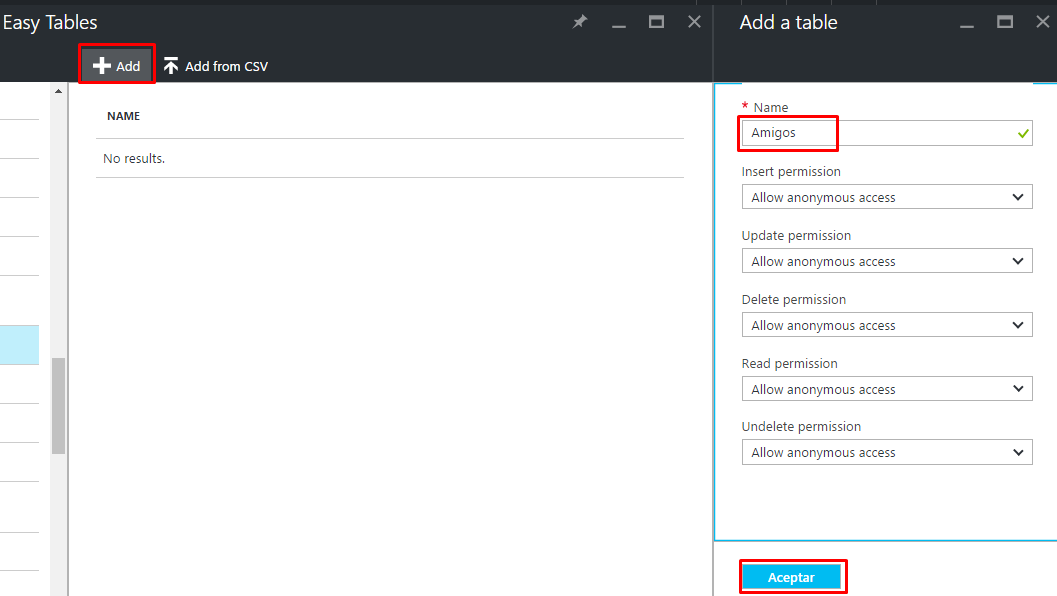
**Paso 18:** En este paso, el servicio móvil busca una cadena de conexión con el nombre **MS\_TableConnectionString**. En caso de ser encontrada, se muestra el mensaje **You already have a data connection**. Simplemente resta inicializar el App Service para que pueda utilizar Easy Tables **marcando el checkbox** que aparece en la sección 2. Da clic en **Initialize App** una vez marcado el checkbox.



**Paso 19:** Recibirás una notificación una vez que el backend haya sido inicializado.

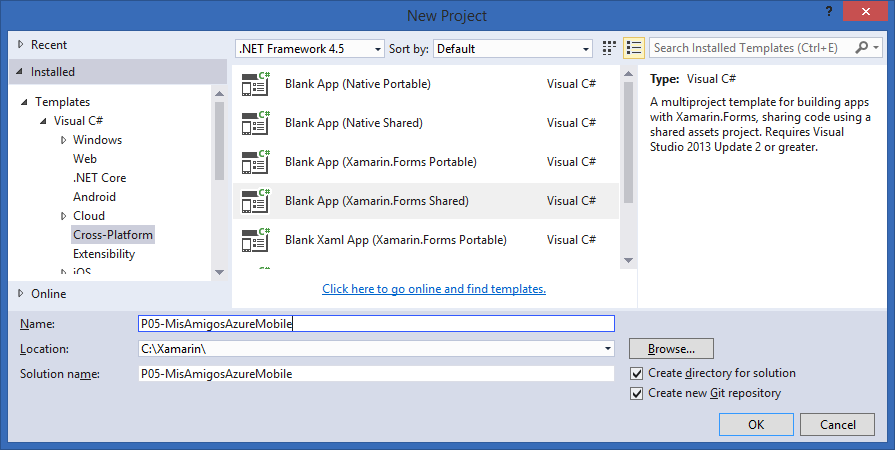


**Paso 20:** Solo resta un paso en el backend. En la sección Easy Tables da clic en el botón **Add** y crea la tabla **Amigos**, dejando los permisos predeterminados y dando clic en **Aceptar**.

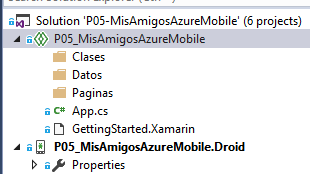


**¡Felicidades! Ya tienes un backend que puedes consumir desde cualquier tipo de aplicación y lenguaje. A continuación mostraremos cómo acceder a él dese una app de Xamarin.**

**Paso 21:** Abre Visual Studio 2015. Crea el proyecto **P05-MisAmigosAzureMobile** que es de tipo Xamarin.Forms Shared (localizado bajo la categoría Cross-Platform):

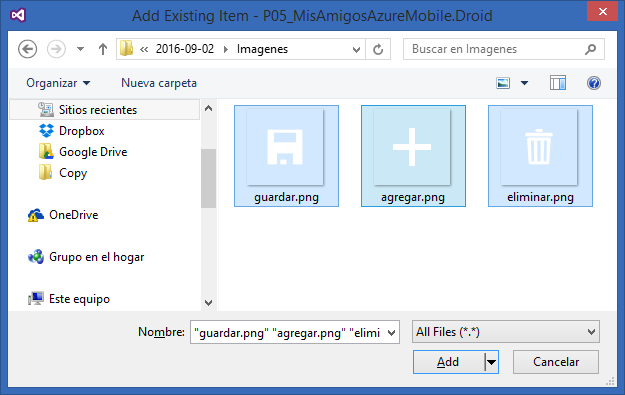


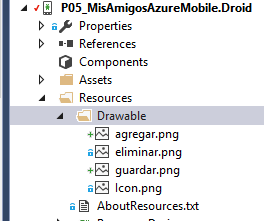
**Paso 22:** Agrega carpetas al proyecto compartido. Da clic derecho sobre el nombre del proyecto compartido (**P05\_MisAmigosAzureMobile**) en el Explorador de Soluciones y selecciona la opción Agregar 🡪 Nueva carpeta. Las carpetas que agregaremos serán: **Clases**, **Paginas** y **Datos**.



**Paso 23:** Agrega imágenes a cada Proyecto que deseas implementar:

a) Para **Android**: En el proyecto **P05\_MisAmigosAzureMobile.Droid**, expande **Resources** y da clic derecho en la carpeta **Drawable**. Selecciona la opción **Agregar** 🡪 **Elemento existente**. Selecciona las 3 imágenes que se incluyen en la práctica, correspondientes los íconos **Agregar, Modificar** y **Eliminar**. Agrégalas.

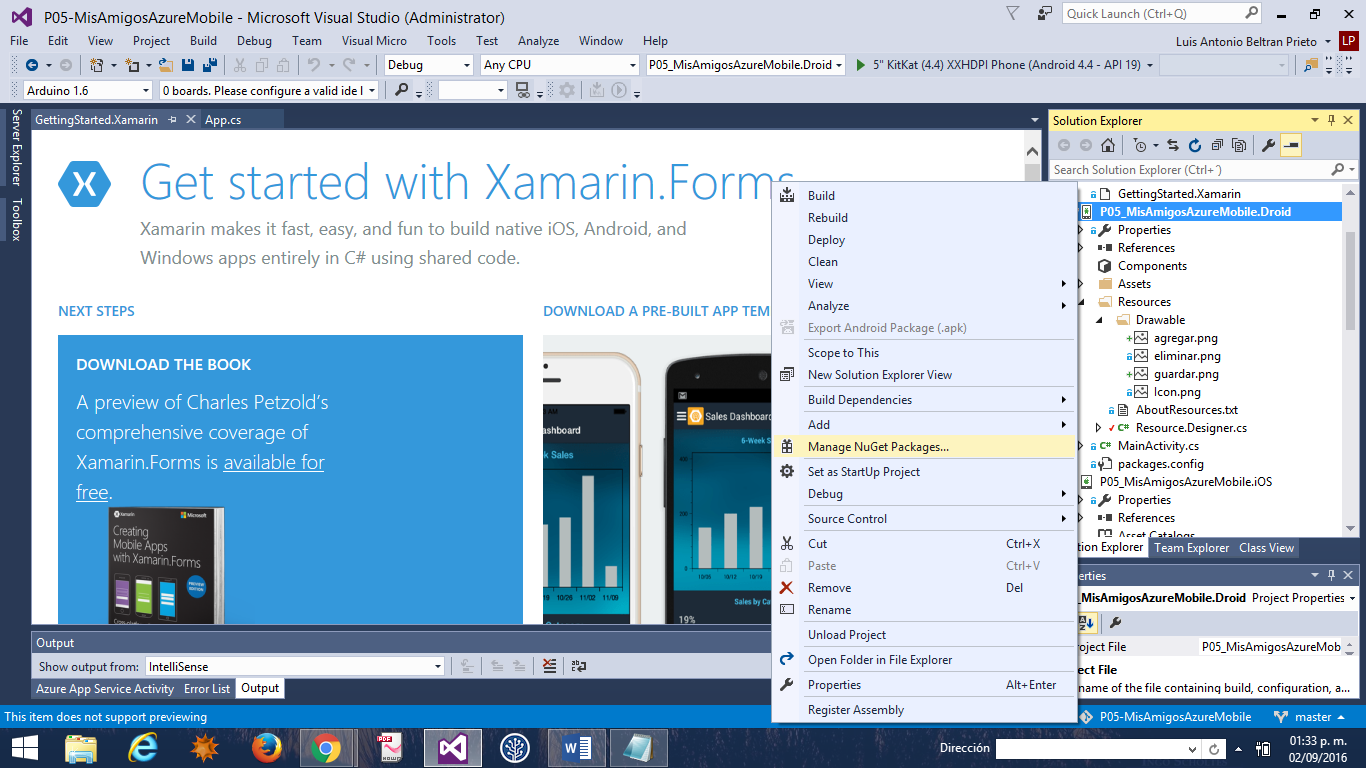




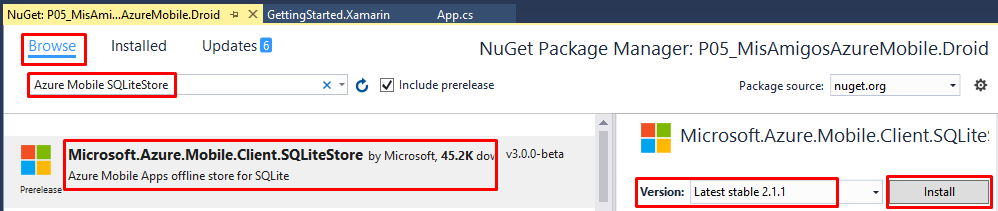
b) Para **iOS**: En el proyecto **P05\_MisAmigosAzureMobile.iOS**, da clic derecho en la carpeta **Resources** y selecciona la opción **Agregar 🡪 Elemento existente**. Selecciona las 3 imágenes que se incluyen en la práctica, correspondientes los íconos **Agregar, Modificar** y **Eliminar**. Agrégalas.

c) Para **Windows Phone, Windows y UWP**: Da clic derecho en el nombre del proyecto respectivo y selecciona la opción **Agregar 🡪 Elemento existente**. Selecciona las 3 imágenes que se incluyen en la práctica, correspondientes los íconos **Agregar, Modificar** y **Eliminar**. Agrégalas.

**Paso 24:** En cada proyecto que desees implementar, agrega paquetes Nuget. Da clic derecho en el nombre del proyecto y selecciona **Manage Nuget Packages**.

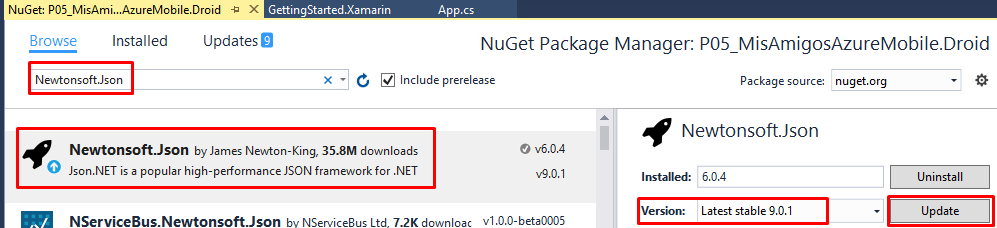


**Paso 25:** Selecciona la pestaña **Browse** e ingresa el nombre **Azure Mobile SQLiteStore**. Selecciona el paquete **Microsoft.Azure.Mobile.Client.SQLiteStore** y del lado derecho elige **la última versión estable** (en este momento es la **versión 2.1.1**). Da clic en **Install**.

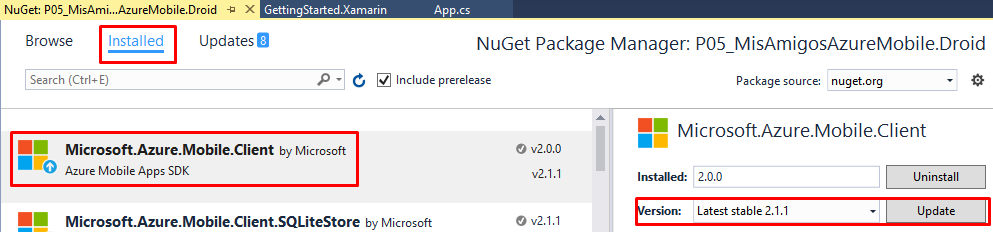


Acepta los cuadros de diálogo que aparezcan (instalación de prerrequisitos y aceptación de términos de licencias).

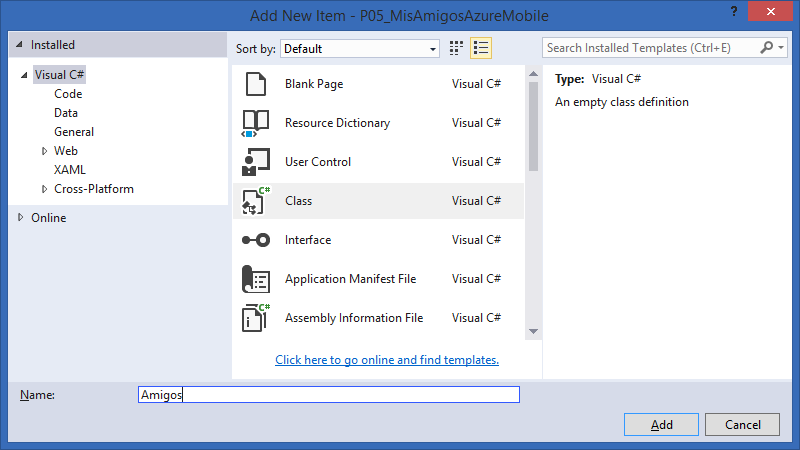
**Paso 26.** Ahora busca e instala/actualiza el paquete **Newtonsoft.Json**. Se recomienda utilizar la última versión estable disponible.



**Paso 27.** Ahora accede a la pestaña **Installed**. Actualiza el paquete **Microsoft.Azure.Mobile.Client** a la versión **2.1.1**, que corresponde con la misma versión del paquete previamente instalado **Microsoft.Azure.Mobile.Client.SQLiteStore.**



**Paso 28:** Vamos al proyecto compartido **(P05\_MisAmigosAzureMobile).** En la carpeta **Clases**,agrega una clase llamada **Amigos**.



**Paso 29:** El código de la clase Amigos es:

using System;

using System.Collections.Generic;

using System.Text;

namespace P05\_MisAmigosAzureMobile.Clases

{

public class Amigos

{

[Newtonsoft.Json.JsonProperty("Id")]

public string Id { get; set; }

[Microsoft.WindowsAzure.MobileServices.Version]

public string AzureVersion { get; set; }

public string Nombre { get; set; }

public int Sexo { get; set; }

public DateTime Cumple { get; set; }

public string Correo { get; set; }

public string Telefono { get; set; }

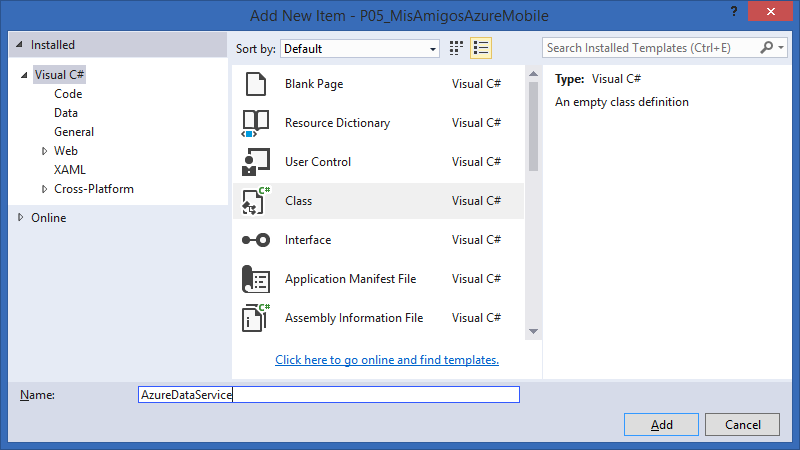
public bool EsMejorAmigo { get; set; }

}

}

Esta clase es el modelo de la tabla Amigos que tenemos en nuestra base de datos en la nube. Cuando la app móvil sea ejecutada por primera vez, esta estructura (modelo) será pasado a la tabla remota para crear los campos indicados en las propiedades. Las 2 primeras son propias de una tabla de SQL Azure, mientras que el resto son elementos inherentes al propósito de la tabla (el tipo de información que queremos almacenar de un amigo).

**Paso 30:** En la carpeta **Datos**, agrega una clase llamada **AzureDataService**.



**Paso 31:** El propósito de esta clase es **conectarnos al backend** creado previamente, además de implementar las **operaciones CRUD** (Create, Read, Update y Delete) de la tabla **Amigos**. En el código que se muestra a continuación revisa el método **Initialize**:

* Si creaste un backend en Azure (pasos 1-20), modifica **la url** del constructor del **MobileServiceClient** por la que obtuviste en el paso 9.
* Si no creaste el backend, deja el código como está. La url que se muestra es la que cree para esta práctica.

using System;

using Microsoft.WindowsAzure.MobileServices;

using Microsoft.WindowsAzure.MobileServices.Sync;

using System.Threading.Tasks;

using System.Collections.Generic;

using Microsoft.WindowsAzure.MobileServices.SQLiteStore;

using P05\_MisAmigosAzureMobile.Clases;

using System.Linq;

namespace P05\_MisAmigosAzureMobile.Datos

{

public class AzureDataService

{

public MobileServiceClient MobileService { get; set; }

IMobileServiceSyncTable<Amigos> tablaAmigos;

bool isInitialized;

public async Task Initialize()

{

if (isInitialized)

return;

MobileService = new MobileServiceClient("http://mis-amigos-app.azurewebsites.net");

const string path = "syncstore-amigos.db";

var store = new MobileServiceSQLiteStore(path);

store.DefineTable<Amigos>();

await MobileService.SyncContext.InitializeAsync(store, new MobileServiceSyncHandler());

tablaAmigos = MobileService.GetSyncTable<Amigos>();

isInitialized = true;

}

public async Task<IEnumerable<Amigos>> ObtenerAmigos()

{

await Initialize();

await SyncAmigos();

return await tablaAmigos.OrderBy(a => a.Nombre).ToEnumerableAsync();

}

public async Task<Amigos> ObtenerAmigo(string id)

{

await Initialize();

await SyncAmigos();

return (await tablaAmigos.Where(a => a.Id == id).Take(1).ToEnumerableAsync()).FirstOrDefault();

}

public async Task<Amigos> AgregarAmigo(string nombre, int sexo, DateTime cumple, string correo, string telefono, bool esMejorAmigo)

{

await Initialize();

var item = new Amigos

{

Nombre = nombre,

Sexo = sexo,

Cumple = cumple,

Correo = correo,

Telefono = telefono,

EsMejorAmigo = esMejorAmigo

};

await tablaAmigos.InsertAsync(item);

await SyncAmigos();

return item;

}

public async Task<Amigos> ModificarAmigo(string id, string nombre, int sexo, DateTime cumple, string correo, string telefono, bool esMejorAmigo)

{

await Initialize();

var item = await ObtenerAmigo(id);

item.Nombre = nombre;

item.Sexo = sexo;

item.Cumple = cumple;

item.Correo = correo;

item.Telefono = telefono;

item.EsMejorAmigo = esMejorAmigo;

await tablaAmigos.UpdateAsync(item);

await SyncAmigos();

return item;

}

public async Task EliminarAmigo(string id)

{

await Initialize();

var item = await ObtenerAmigo(id);

await tablaAmigos.DeleteAsync(item);

await SyncAmigos();

}

public async Task SyncAmigos()

{

await tablaAmigos.PullAsync("Amigos", tablaAmigos.CreateQuery());

await MobileService.SyncContext.PushAsync();

}

}

}

**Paso 32:** En la clase **App**, creamos un objeto estático de la clase **AzureDataService** y lo identificaremos simplemente como **AzureService**. También modificamos de una vez la página de inicio (**MainPage**) a través de una nueva instancia de **NavigationPage** al objeto **MainPage**. Esto permite que nuestra aplicación maneje más de una página y se comparta información entre páginas utilizando un esquema de Stack, es decir, cada nueva página se muestra encima de la anterior (push). Al presionar el botón de atrás, se puede regresar a la última página visitada (pop). A su vez, el NavigationPage requiere indicar la página específica que se cargará al principio. Utiliza una nueva instancia de **PaginaListaAmigos** (agrega los espacios de nombre indicados).

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using Xamarin.Forms;

using P05\_MisAmigosAzureMobile.Datos;

using P05\_MisAmigosAzureMobile.Paginas;

namespace P05\_MisAmigosAzureMobile

{

public class App : Application

{

public static AzureDataService AzureService;

public App ()

{

AzureService = new AzureDataService();

MainPage = new NavigationPage(new PaginaListaAmigos());

}

protected override void OnStart ()

{

// Handle when your app starts

}

protected override void OnSleep ()

{

// Handle when your app sleeps

}

protected override void OnResume ()

{

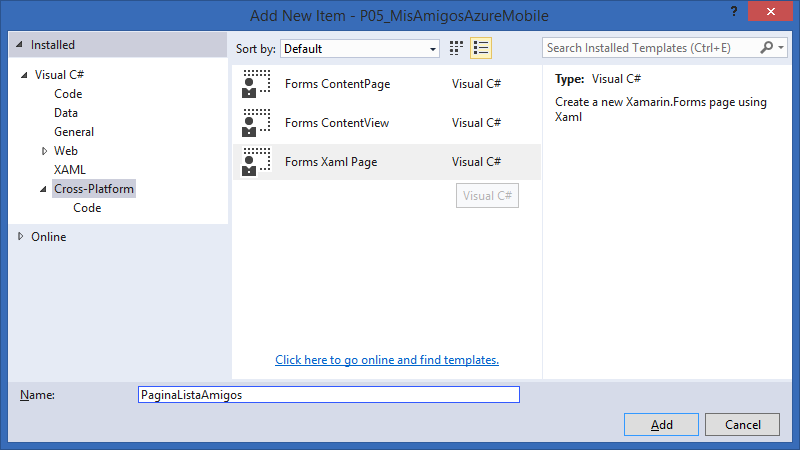
// Handle when your app resumes

}

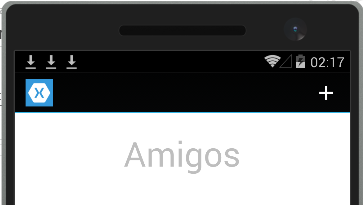
}

}

**Paso 33**. Agrega otra página dando clic derecho sobre la carpeta **Paginas** y selecciona **Agregar 🡪 Nuevo elemento**. Selecciona **Forms Xaml Page** (en la categoría **Cross-Platform**) y escribe el nombre **PaginaListaAmigos**:



**Paso 34.** La interfaz de usuario de esta página es muy sencilla aparentemente, pues consiste en un **Label** (título: Amigos), seguido de un **ListView** que mostrará la **lista de amigos almacenados** en la base de datos y un **ToolbarItem** (botón) en la barra de herramientas que permitirá **agregar** nuevos amigos a la colección llamando a la página anterior. La interfaz se muestra a continuación, seguida del código XAML respectivo:



<?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="P05\_MisAmigosAzureMobile.Paginas.PaginaListaAmigos">

<ContentPage.Content>

<StackLayout BackgroundColor="White">

<Label Text="Amigos" FontSize="40" HorizontalOptions="Center"/>

<ListView x:Name="lsvAmigos" ItemSelected="lsvAmigos\_Selected" RowHeight="50">

<ListView.ItemTemplate>

<DataTemplate>

<ViewCell>

<StackLayout VerticalOptions="FillAndExpand" HorizontalOptions="FillAndExpand">

<StackLayout Orientation="Horizontal">

<Label Text="{Binding Nombre}"

TextColor="Blue" FontSize="16"/>

<Switch IsToggled="{Binding EsMejorAmigo}" HorizontalOptions="EndAndExpand"/>

</StackLayout>

<StackLayout Orientation="Horizontal">

<Label Text="{Binding Telefono}" HorizontalOptions="StartAndExpand" TextColor="Green" FontSize="12"/>

<Label Text="{Binding Correo}" HorizontalOptions="EndAndExpand" TextColor="Green" FontSize="12"/>

</StackLayout>

</StackLayout>

</ViewCell>

</DataTemplate>

</ListView.ItemTemplate>

</ListView>

</StackLayout>

</ContentPage.Content>

<ContentPage.ToolbarItems>

<ToolbarItem x:Name="btnNuevo" Text="Nuevo" Priority="0" Clicked="btnNuevo\_Click" Order="Primary" Icon="agregar.png"/>

</ContentPage.ToolbarItems>

</ContentPage>

La “complejidad” de la interfaz está en el elemento **DataTemplate** del ListView. Esta propiedad es una plantilla que representa el aspecto que **cada elemento** tendrá, además de la información a mostrar a través del enlace a datos (**Binding**). Por ejemplo, se mostrará el nombre, correo y teléfono en un Label junto con un Switch que indica si es un mejor amigo. Estas propiedades se obtienen de un modelo (clase) que no hemos creado aún.

Los elementos del ListView serán suministrados utilizando la propiedad **ItemsSource**.

**Paso 35.** Accede al código de **PaginaListaAmigos.xaml.cs**. Este archivo representa el “code-behind” de la página. En esta página vamos a:

* **Mostrar la lista de amigos al cargar la página** (se sobreescribe el método **OnAppearing** para asegurar que el control ha sido inicializado).
* **Navegar a PaginaAmigos.xaml cuando se seleccione un amigo de la lista**. Se debe enviar la información de qué amigo fue seleccionado.
* **Navegar a PaginaAmigos.xaml cuando se presione el botón Nuevo**. No se envía otra información.

El código completo es el siguiente:

using System;

using Xamarin.Forms;

using P05\_MisAmigosAzureMobile.Clases;

namespace P05\_MisAmigosAzureMobile.Paginas

{

public partial class PaginaListaAmigos : ContentPage

{

public PaginaListaAmigos ()

{

InitializeComponent ();

}

protected async override void OnAppearing()

{

base.OnAppearing();

lsvAmigos.ItemsSource = await App.AzureService.ObtenerAmigos();

}

private void lsvAmigos\_Selected(object sender, SelectedItemChangedEventArgs e)

{

if (e.SelectedItem != null)

{

Amigos amigo = e.SelectedItem as Amigos;

PaginaAmigo pagina = new PaginaAmigo();

pagina.ID = amigo.Id;

Navigation.PushAsync(pagina);

}

}

void btnNuevo\_Click(object sender, EventArgs a)

{

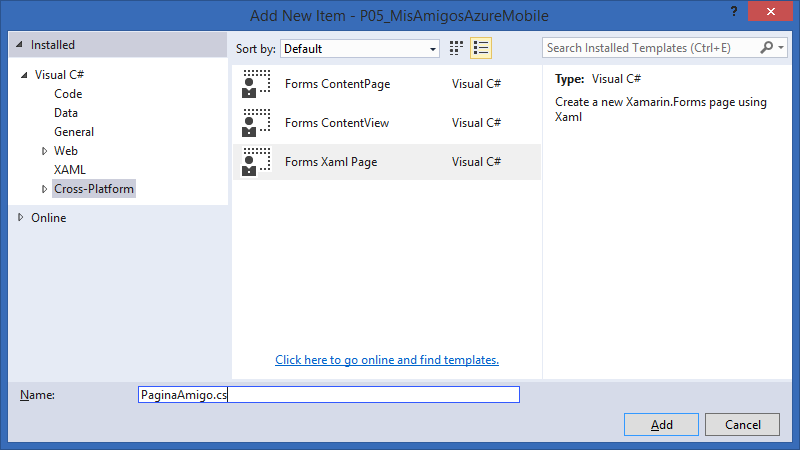
Navigation.PushAsync(new PaginaAmigo());

}

}

}

**Paso 36:** En la carpeta **Paginas**, agrega un nuevo **Forms Xaml Page** (en la categoría **Cross-Platform**) para crear una nueva página que incluya diseño (**XAML**) y lógica (**C#**), escribe el nombre **PaginaAmigo**:



**Paso 37:** En esta página vamos a mostrar al usuario la información que queremos almacenar de un amigo, para lo cual construiremos la siguiente interfaz:



* Para el **nombre** utilizaremos un View de tipo **Entry**, que es básicamente una caja de texto con la sugerencia (*hint* o *placeholder*) “Nombre”
* Para el **sexo** (Hombre/Mujer) utilizamos un **Picker**, que permite elegir entre varias opciones
* Para el **cumpleaños** utilizamos un **DatePicker** para que el usuario seleccione una fecha y la muestre en el formato [día] de [mes]
* Para el **correo** y **teléfono**, se utilizan 2 **Entry**.
* Finalmente, se puede indicar si es o no un **mejor amigo** mediante un control **Switch**.

Observa también que en la parte superior aparecen 2 íconos. En realidad son ToolbarItems que se agregan a la Toolbar de la página. La posición de la barra de herramientas es una característica dependiente de plataforma (por ejemplo, en Android aparecen en la parte superior, mientras que en Windows Phone aparecen en la parte inferior).

El código **XAML** respectivo se muestra a continuación.

<?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="P05\_MisAmigosAzureMobile.Paginas.PaginaAmigo">

<ContentPage.Content>

<StackLayout Spacing="20" Padding="20">

<Entry x:Name="txtNombre" Placeholder="Nombre" />

<Picker x:Name="pckSexo" Title="Sexo">

<Picker.Items>

<x:String>Hombre</x:String>

<x:String>Mujer</x:String>

</Picker.Items>

</Picker>

<StackLayout Orientation="Horizontal">

<Label Text="Cumpleaños" VerticalOptions="CenterAndExpand"/>

<DatePicker x:Name="dppCumple" Format="dd \de MMMM" VerticalOptions="CenterAndExpand" HeightRequest="50"/>

</StackLayout>

<Entry x:Name="txtCorreo" Placeholder="Correo" />

<Entry x:Name="txtTelefono" Placeholder="Telefono" />

<StackLayout Orientation="Horizontal">

<Label Text="¿Es mejor amigo?" VerticalOptions="CenterAndExpand"/>

<Switch x:Name="swtMejorAmigo" VerticalOptions="CenterAndExpand"/>

</StackLayout>

</StackLayout>

</ContentPage.Content>

<ContentPage.ToolbarItems>

<ToolbarItem x:Name="btnGuardar" Text="Guardar" Priority="0" Clicked="btnGuardar\_Click" Order="Primary" Icon="guardar.png"/>

<ToolbarItem x:Name="btnEliminar" Text="Eliminar" Priority="1" Clicked="btnEliminar\_Click" Order="Primary" Icon="eliminar.png"/>

</ContentPage.ToolbarItems>

</ContentPage>

**Paso 38.** Accede al código de **PaginaAmigo.xaml.cs**. Este archivo representa el “code-behind” de la página. Aquí vamos a llamar los métodos creados en el paso 31 para agregar, modificar o eliminar amigos a través de nuestra conexión AzureService. En esta página se necesita:

* **Cargar la información del amigo seleccionado** (se utiliza el método OnAppering para asegurar que los controles han sido dibujados).
* **Guardar el amigo con la información proporcionada** al presionar el botón Guardar.
* **Eliminar al amigo seleccionado** al presionar el botón Eliminar.

using System;

using Xamarin.Forms;

using P05\_MisAmigosAzureMobile.Clases;

namespace P05\_MisAmigosAzureMobile.Paginas

{

public partial class PaginaAmigo : ContentPage

{

public string ID = "";

protected async override void OnAppearing()

{

base.OnAppearing();

if (ID != "")

{

Amigos amigo = await App.AzureService.ObtenerAmigo(ID);

txtNombre.Text = amigo.Nombre;

pckSexo.SelectedIndex = amigo.Sexo;

dppCumple.Date = amigo.Cumple;

txtCorreo.Text = amigo.Correo;

txtTelefono.Text = amigo.Telefono;

swtMejorAmigo.IsToggled = amigo.EsMejorAmigo;

}

}

void btnGuardar\_Click(object sender, EventArgs a)

{

string nombre = txtNombre.Text;

int sexo = pckSexo.SelectedIndex;

DateTime cumple = dppCumple.Date;

string correo = txtCorreo.Text;

string telefono = txtTelefono.Text;

bool esMejorAmigo = swtMejorAmigo.IsToggled;

if (ID == String.Empty)

App.AzureService.AgregarAmigo(nombre, sexo, cumple, correo, telefono, esMejorAmigo);

else

App.AzureService.ModificarAmigo(ID, nombre, sexo, cumple, correo, telefono, esMejorAmigo);

Navigation.PopAsync();

}

void btnEliminar\_Click(object sender, EventArgs a)

{

if (ID != "")

{

App.AzureService.EliminarAmigo(ID);

Navigation.PopAsync();

}

}

public PaginaAmigo ()

{

InitializeComponent ();

}

}

}

**Paso 39.** Ahora ve al archivo **MainActivity** que está en el proyecto de **Android** (**P05\_MisAmigosAzureMobile.Droid**). Simplemente tienes que agregar una llamada al método **Init** de los servicios móviles de **Azure** dentro del método **OnCreate**:

protected override void OnCreate (Bundle bundle)

{

base.OnCreate (bundle);

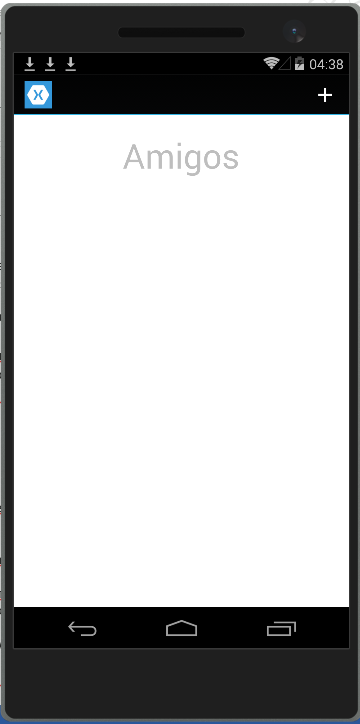
global::Xamarin.Forms.Forms.Init (this, bundle);

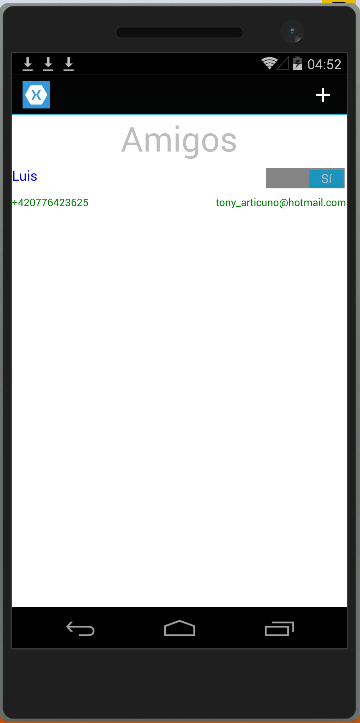
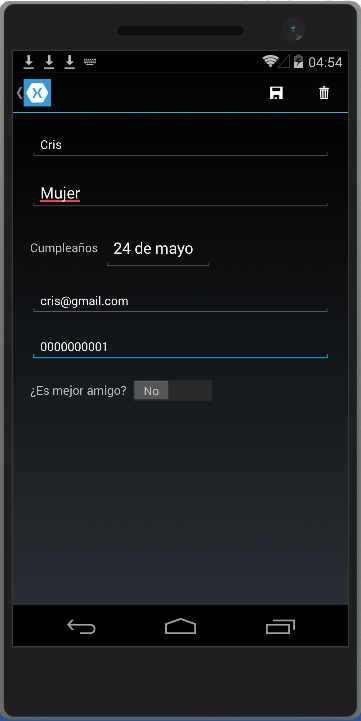
Microsoft.WindowsAzure.MobileServices.CurrentPlatform.Init();

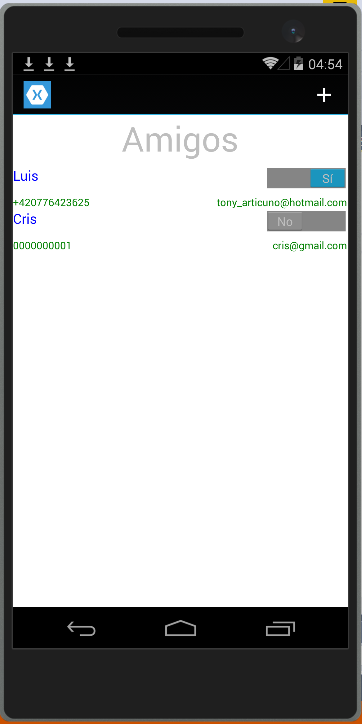
LoadApplication(new P05\_MisAmigosAzureMobile.App ());

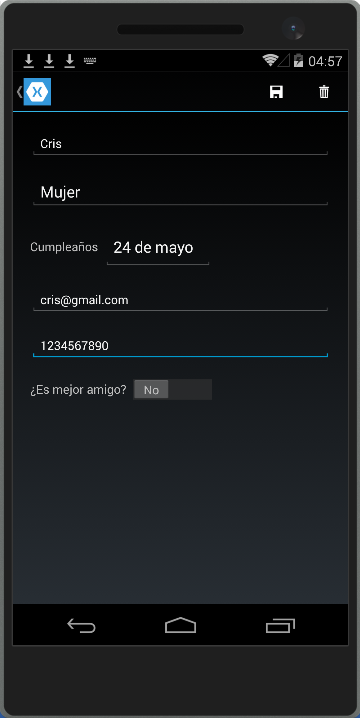
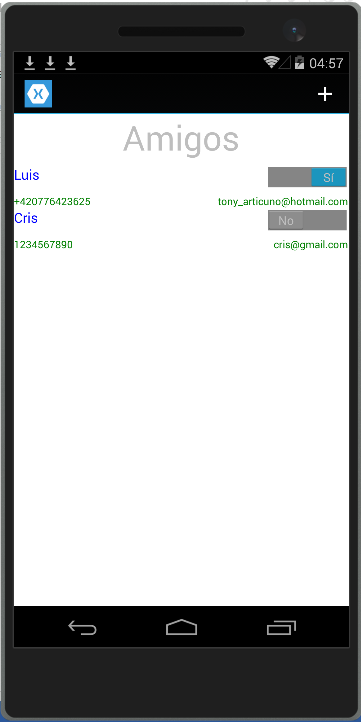
}

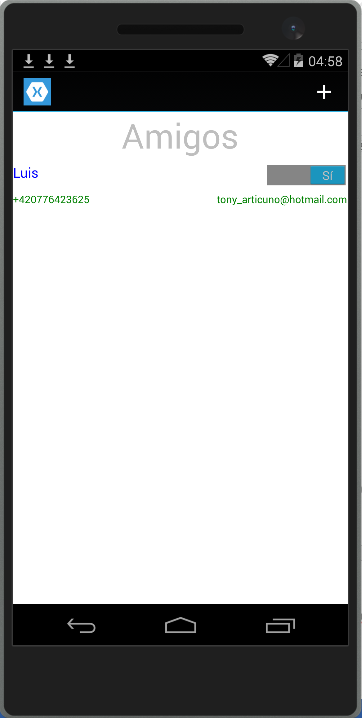
**Paso 40:** ¡Listo! Compila y ejecuta la aplicación. Revisa su correcto funcionamiento.

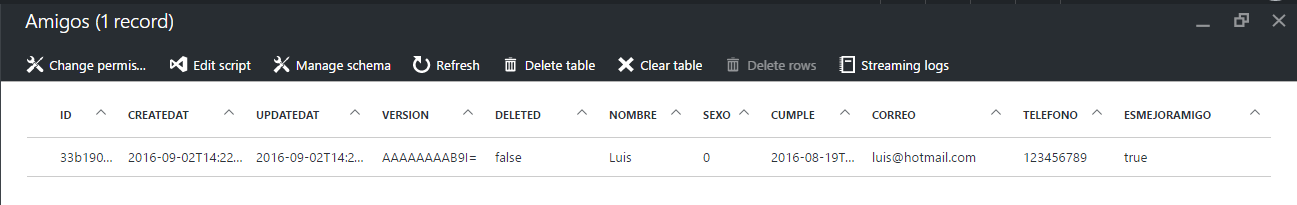
 



Después de haber insertado el primer registro, checa la tabla Amigos en el portal de Azure. Se ha actualizado la estructura de esta tabla:



**¡Felicidades! Has creado una aplicación móvil multiplataforma que almacena información en una base de datos en la nube. El backend de Azure puede ser consumido desde otra aplicación y en cualquier otro lenguaje.**