

Autenticación con Asp.Net Core Identity

Seguridad Aplicaciones Asp.Net

Introducción

El marco ASP.NET
Core Identity

Registro de
usuarios

Inicio y cierre de
sesión

Cómo
personalizar
ASP.NET Core
Identity

Bloqueo de
usuario

Tokens

Autenticación en
dos fases

Proveedores de
inicio de sesión
externos

Introducción



- Autenticación
- ASP.NET Core Identity
- Servidor de identidades

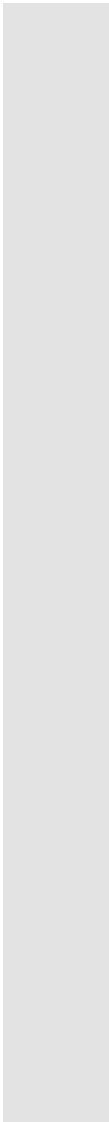


- Autorización
- Autorización ASP.NET Core



Asp.Net Core Identity

Un marco de desarrollo que ayuda a gestionar y autenticar usuarios en aplicaciones



Características

Inicio y cierre
de sesión

Registro

Inicios de
sesión
externos

Administración
de contraseñas

Bloqueo

Autenticación
en dos fases

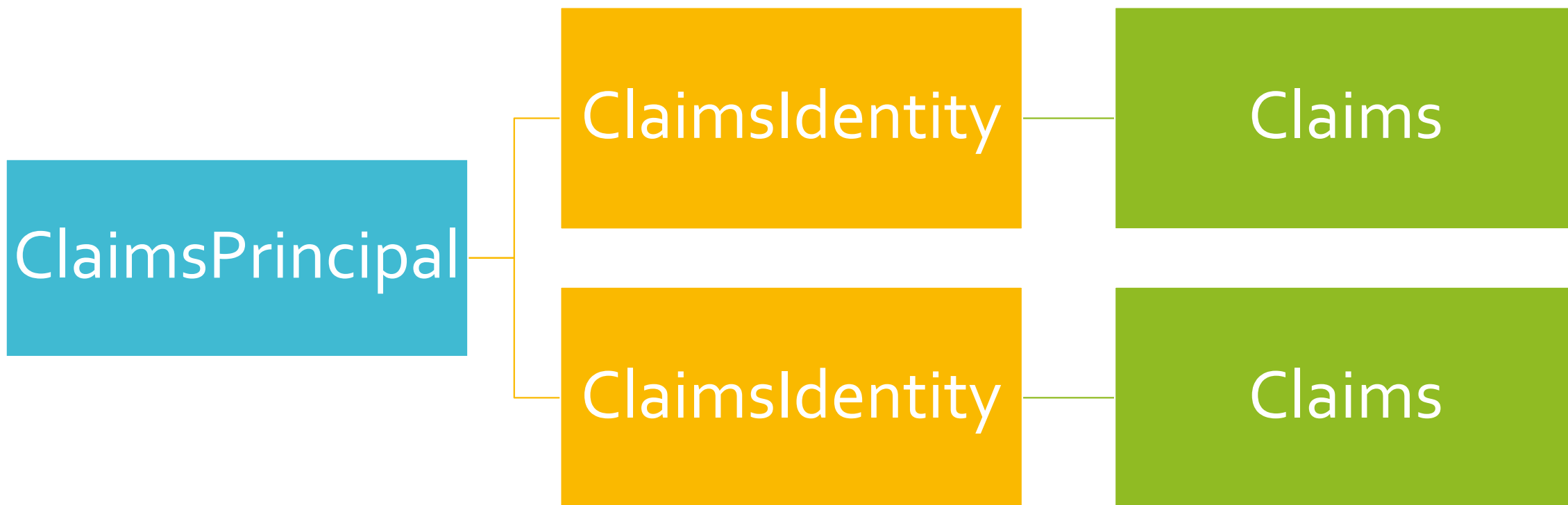
Identidades
basadas en
Claims

Nombre

Ciudad

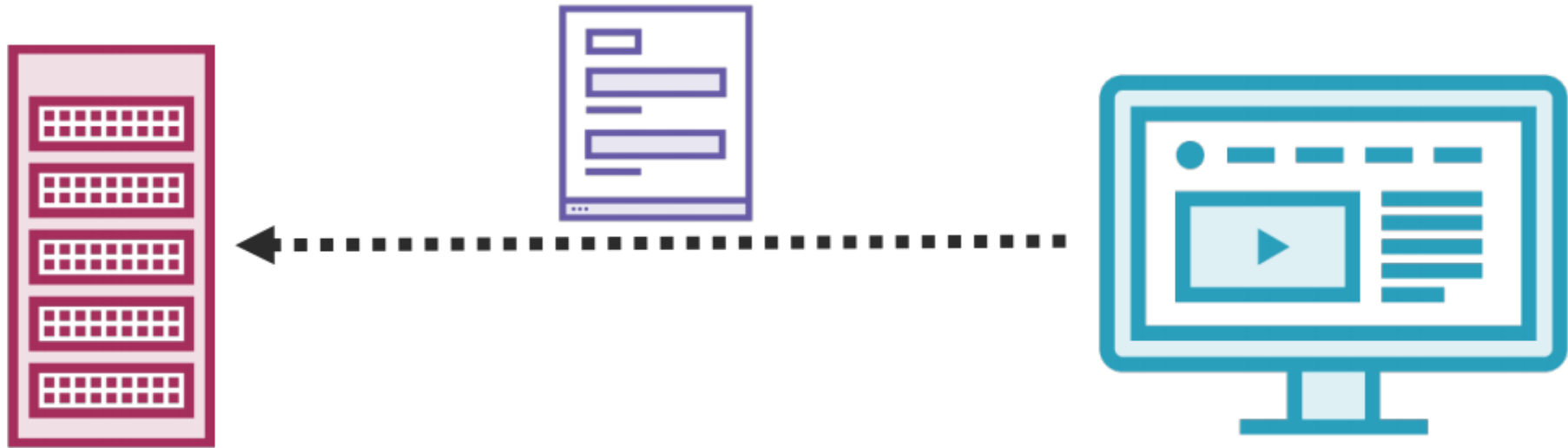
Número de
empleado

Nivel de
acceso



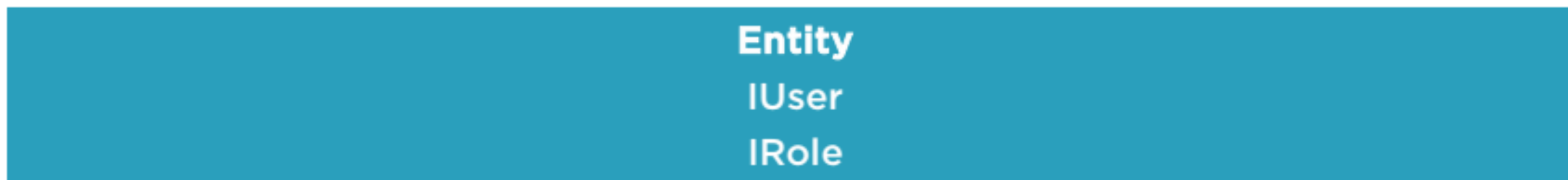
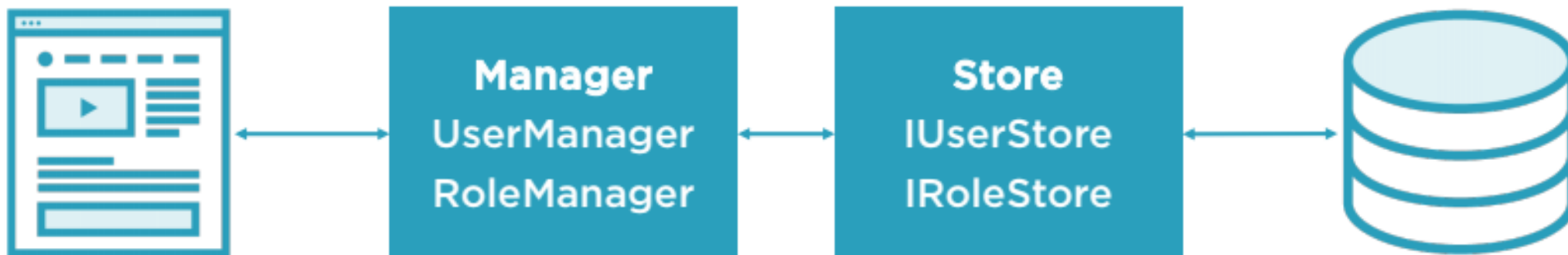
Identities basadas en Claims

La cookie de Identity



La cookie de Identity





Arquitectura Asp.Net Identity

IUser

- No indica cómo se autentican los usuarios
- TKey es un genérico

```
public interface IUser<out TKey>
{
    TKey Id { get; }
    string UserName { get; set; }
}
```

IUserStore

- Abstracción del almacenamiento de datos de usuarios

```
public interface IUserStore<TUser, in TKey> : IDisposable where TUser
: class, IUser<TKey> {
    Task CreateAsync(TUser user);
    Task UpdateAsync(TUser user);
    Task DeleteAsync(TUser user);
    Task<TUser> FindByIdAsync(TKey userId);
    Task<TUser> FindByNameAsync(string userName);
}
```

IUserEmailStore

```
public interface IUserEmailStore<TUser, in TKey> :  
    IUserStore<TUser, TKey> where TUser : class, IUser<TKey> {  
    Task SetEmailAsync(TUser user, string email);  
    Task<string> GetEmailAsync(TUser user);  
    Task<bool> GetEmailConfirmedAsync(TUser user);  
    Task SetEmailConfirmedAsync(TUser user, bool  
confirmed);  
    Task<TUser> FindByEmailAsync(string email);  
}
```

IUserPhoneNumberStore

```
public interface IUserPhoneNumberStore<TUser, in TKey> {  
    Task SetPhoneNumberAsync(TUser user, string phoneNumber);  
    Task<string> GetPhoneNumberAsync(TUser user);  
    Task<bool> GetPhoneNumberConfirmedAsync(TUser user);  
    Task SetPhoneNumberConfirmedAsync(TUser user, bool confirmed);  
}
```

UserManager

```
public class UserManager<TUser, TKey> : IDisposable
    where TUser : class, IUser<TKey>
    where TKey : IEquatable<TKey> {
        public UserManager(IUserStore<TUser, TKey> store) {
        }
    }
}
```

Demo consola registro y autenticacion

```
class Program
{
    0 referencias
    static void Main(string[] args)
    {
        var username = "scott@scottbrady91.com";
        var password = "Password123!";

        var userStore = new UserStore<IdentityUser>();
        var userManager = new UserManager<IdentityUser>(userStore);

        //var creationResult = userManager.Create(new IdentityUser(username), password);
        //Console.WriteLine("Creation: {0}", creationResult.Succeeded);

        var user = userManager.FindByName(username);
        //var claimResult = userManager.AddClaim(user.Id, new Claim("given_name", "scott"));
        //Console.WriteLine("Claim Added: {0}", claimResult.Succeeded);

        var checkPassword = userManager.CheckPassword(user, password);
        Console.WriteLine("Password Match: {0}", checkPassword);
    }
}
```


Demo sitio web NuGet



Microsoft.AspNet.Identity.Owin por Microsoft
Owin implementation for ASP.NET Identity.

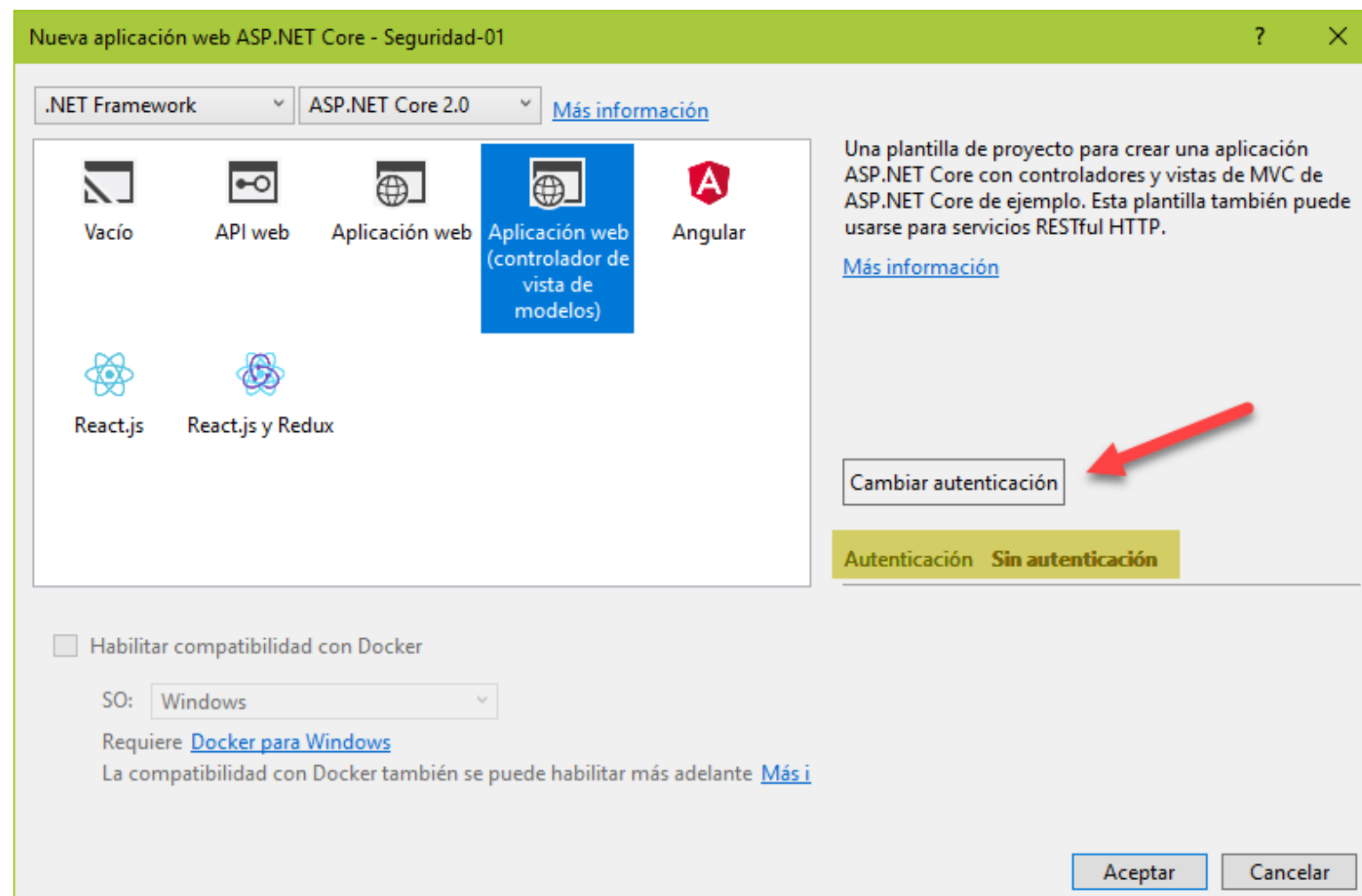


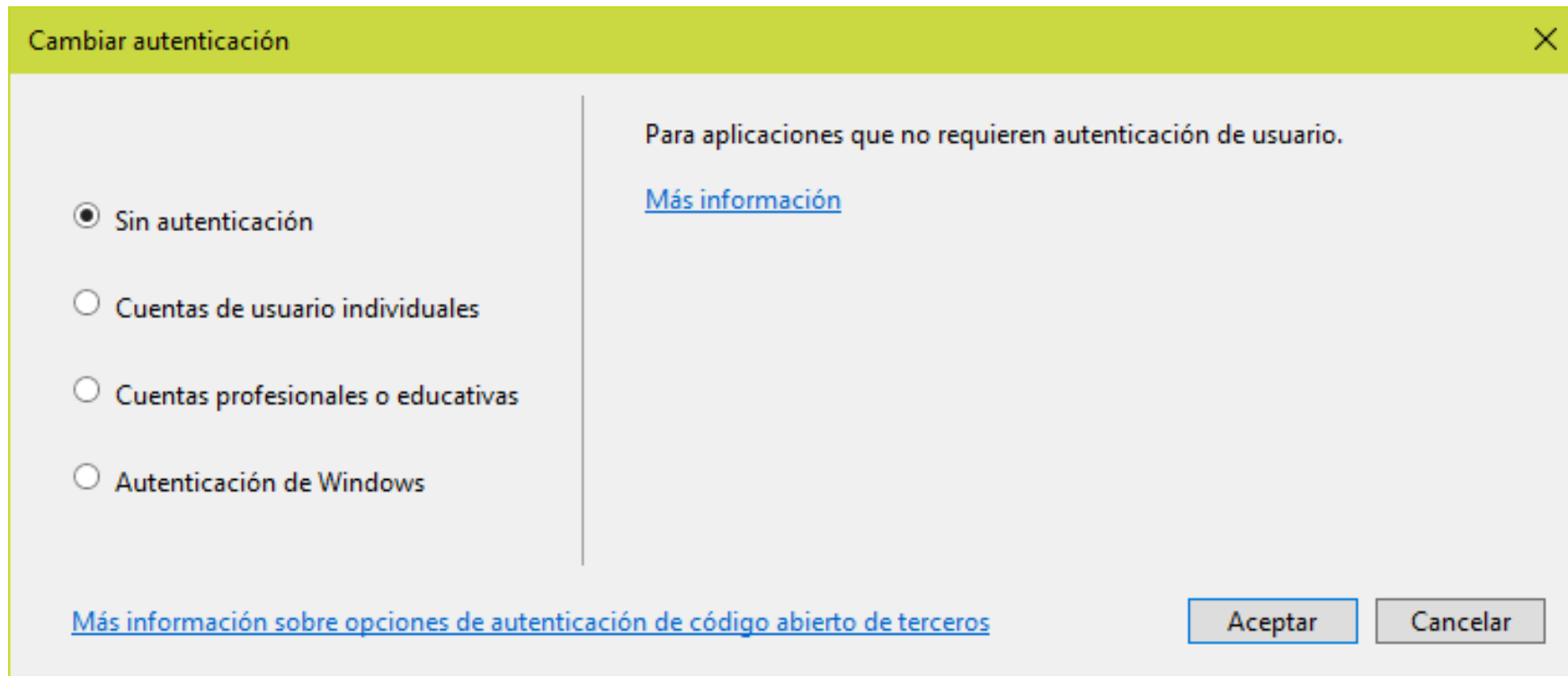
Microsoft.AspNet.Identity.EntityFramework por Microsoft
ASP.NET Identity providers that use Entity Framework.



Microsoft.Owin.Host.SystemWeb por Microsoft
OWIN server that enables OWIN-based applications to run on IIS using the ASP.NET request pipeline.

Opciones de autenticación al crear nuevo proyecto





Sin Autenticación

Cambiar autenticación ✕

☐ Sin autenticación

☒ Cuentas de usuario individuales

☐ Cuentas profesionales o educativas

☐ Autenticación de Windows

Almacenar cuentas de usuario en aplicación ▼ [Más información](#)

Seleccione esta opción para crear un proyecto que incluya un almacén de cu

[Más información sobre opciones de autenticación de código abierto de terceros](#)

Aceptar Cancelar

Cuentas de usuario individuales

Cambiar autenticación

Para aplicaciones que autentican usuarios con Active Directory, Microsoft Azure Active Directory u Office 365.

[Más información](#)

☐ Sin autenticación

☐ Cuentas de usuario individuales

☒ Cuentas profesionales o educativas

☐ Autenticación de Windows

Nube: organización única ⓘ

Dominio: hispafoxgmail.onmicrosoft.com ⓘ

Permisos de acceso de directorio:

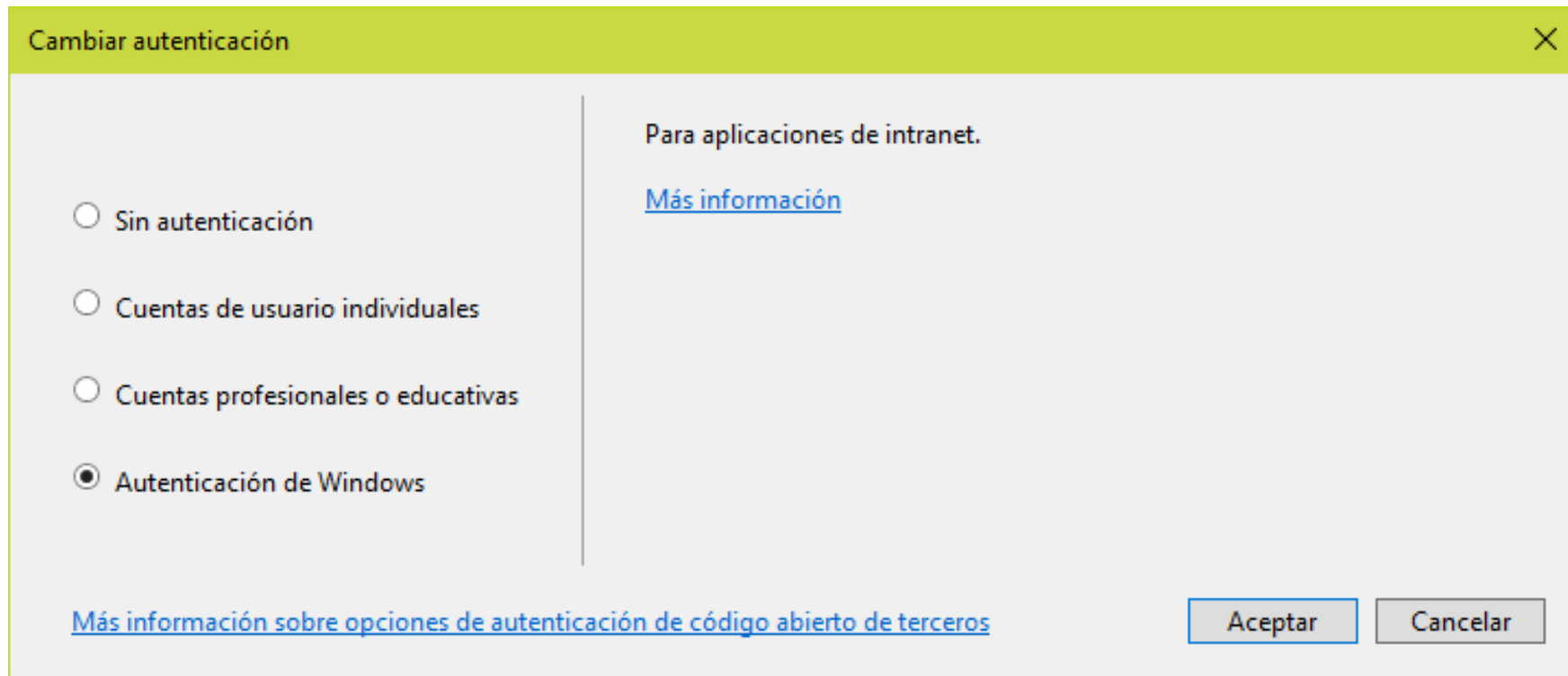
☐ Leer datos de directorio ⓘ

▼ Más opciones

[Más información sobre opciones de autenticación de código abierto de terceros](#)

Aceptar Cancelar

Cuentas profesionales o educativas



Autenticación de Windows

Agregar Identity a un proyecto existente: Identity

- Agregar NuGet: Microsoft.AspNetCore.Identity.EntityFrameworkCore
- En ConfigureServices agregar AddIdentity

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddDbContext<ApplicationDbContext>(options =>
        options.UseSqlServer(Configuration.GetConnectionString("DefaultConnection")));

    services.AddIdentity<ApplicationUser, IdentityRole>()
        .AddEntityFrameworkStores<ApplicationDbContext>()
        .AddDefaultTokenProviders();

    // Add application services.
    services.AddTransient<IEmailSender, EmailSender>();

    services.AddMvc();
}
```

Usuario

- Nombre de usuario
- Email

Claims de Usuario

- Num. Empleado
- Cuenta Corriente

Roles

- Escritor
- Lector

Claims de Roles

- Editor de artículos
- Lector de artículos

Claims

- Nombre de usuario
- Email
- Numero de Empleado
- Cuenta Corriente
- Escritor
- Lector
- Edita Artículos
- Lee Artículos

Lista de Claims

Agregar Identity a un proyecto: Entity Framework

- Agregar NuGet: Microsoft.EntityFrameworkCore.SqlServer
- Agregar AddDbContext
- Agregar UseAuthentication en Configure()
- Crear la base de datos con: dotnet ef migrations add initial (Enable-Migrations)
- Dotnet ef database update (Update-Database)

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddDbContext<ApplicationDbContext>(options =>
        options.UseSqlServer(Configuration.GetConnectionString("DefaultConnection")));

    services.AddIdentity<ApplicationUser, IdentityRole>()
        .AddEntityFrameworkStores<ApplicationDbContext>()
        .AddDefaultTokenProviders();

    // Add application services.
    services.AddTransient<EmailSender, EmailSender>();

    services.AddMvc();
}
```

↑ void ConfigureServices(IServiceCollection)

Cómo agregar funcionalidades al usuario

Agregando propiedades a la clase IdentityUser en lugar de hacerlo como claims separados

Agregando mas entidades al contexto de base de datos con el fin de crear una estructura mas compleja para los datos de usuario

Tipos de Tokens

Confirmación
de cuenta de
correo

Recuperación
de contraseña

Autenticación
en dos fases

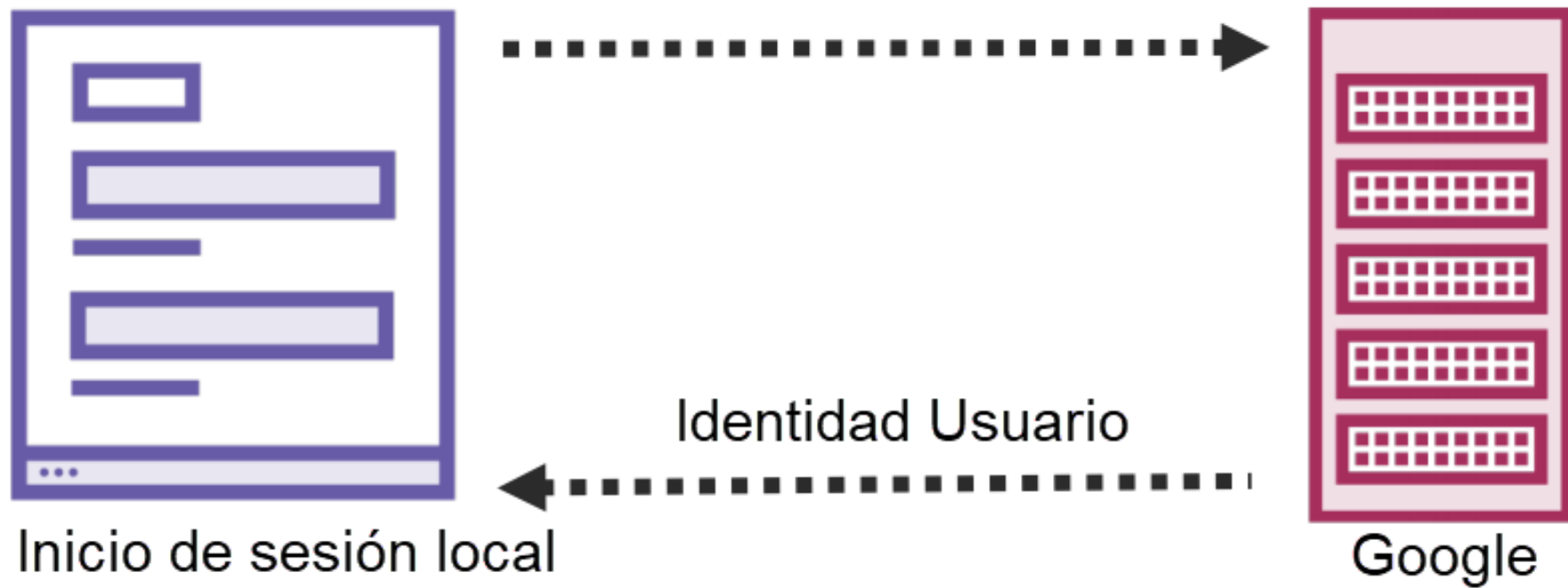
Confirmación
de número de
teléfono

Proveedores de autenticación externos

Proveen al usuario la
posibilidad de iniciar
sesión con una cuenta
externa al sistema

El usuario no tiene
porqué recordar el
usuario y contraseña
de esta aplicación

Inicio de sesión único



Proveedores de autenticación externos

Resumen

Marco basado en
Claims (Notificaciones)
para agregar
autenticación local a
nuestras aplicaciones

La mayor parte de los
componentes pueden
ser reemplazados
(Inyección de
dependencias)

Totalmente
configurable

Soporte para
proveedores externos
de autenticación