

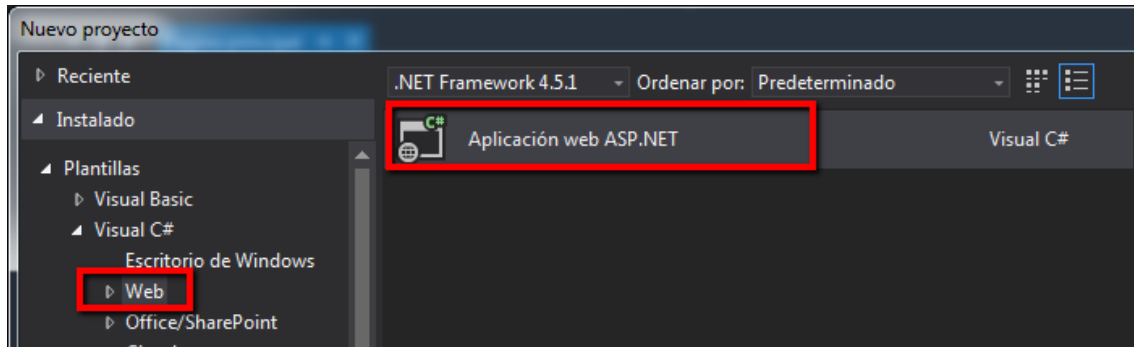
CONTENIDO

Configuración de inicio de sesión con Facebook y google para ASP.NET MVC 5 y Visual Studio 2013	2
Creación del proyecto	2
Habilitando registro con google	5
Habilitando el registro con facebook	9
Datos almacenados	12
Capturando nuevos atributos del cliente	13
Datos Capturados del registro	16
Referencias	17

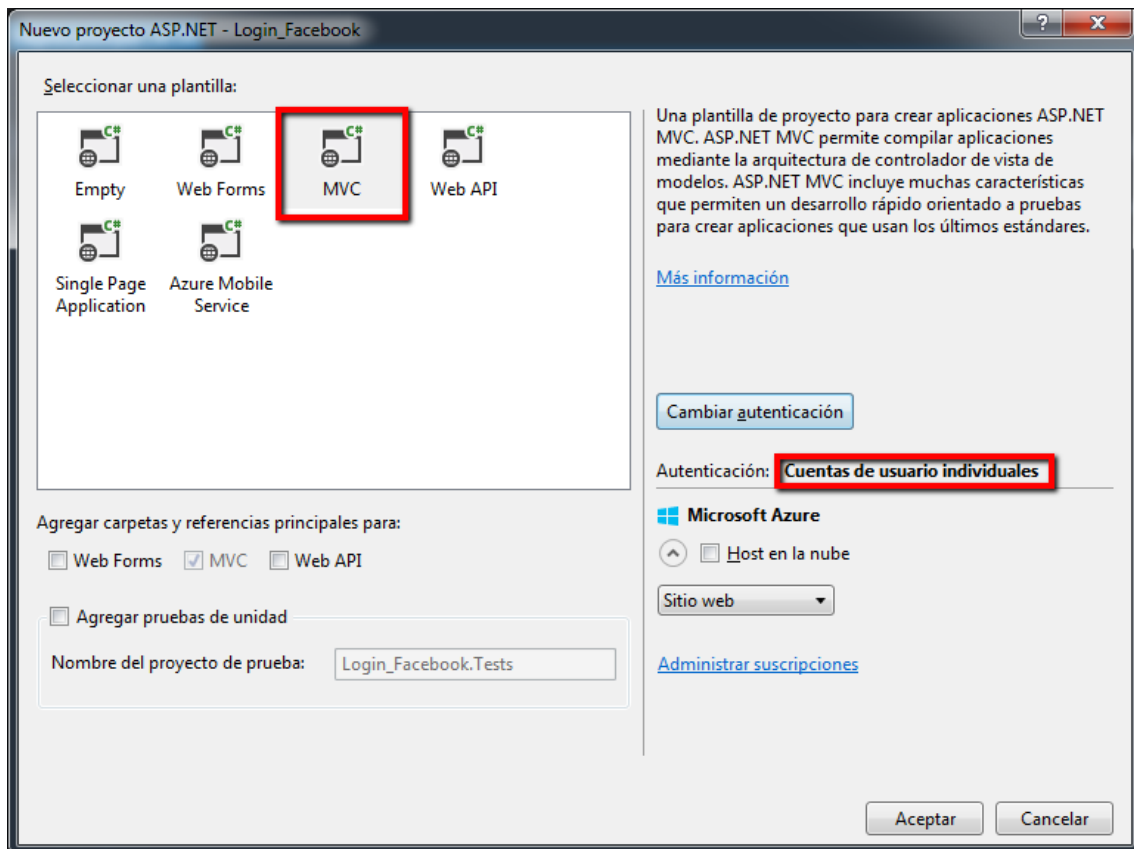
CONFIGURACIÓN DE INICIO DE SESIÓN CON FACEBOOK Y GOOGLE PARA ASP.NET MVC 5 Y VISUAL STUDIO 2013

CREACIÓN DEL PROYECTO

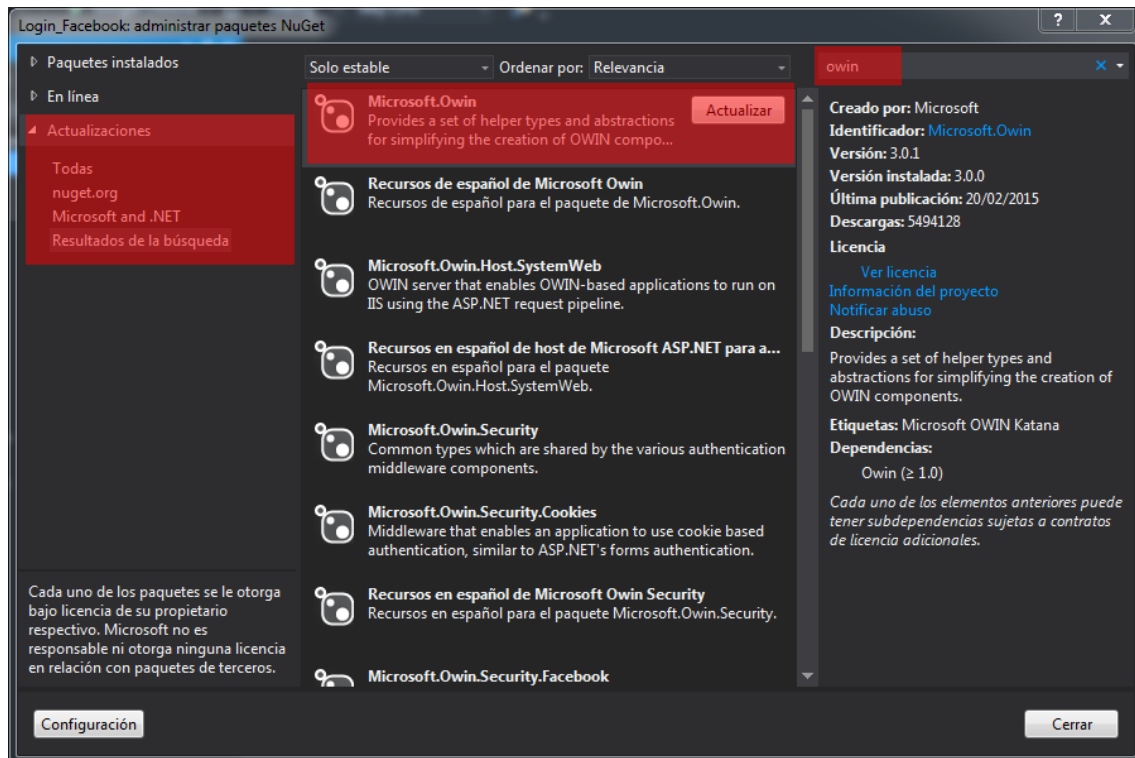
Creamos el proyecto web



Seleccionamos la plantilla "MVC" y cambiamos la autenticación que se nos proporciona a "Cuenta de usuario individuales" para facilitarnos porción de código al implementar el login con Facebook y Gmail.

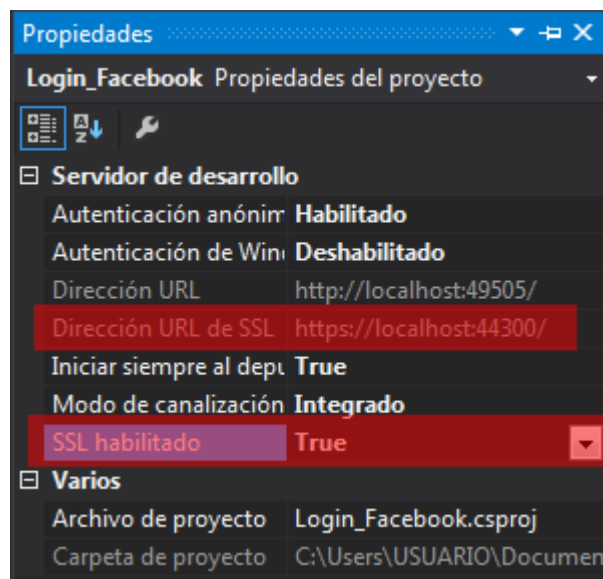


Ingresamos a configurar los paquetes NuGet del proyecto y actualizamos los paquetes OWIN.

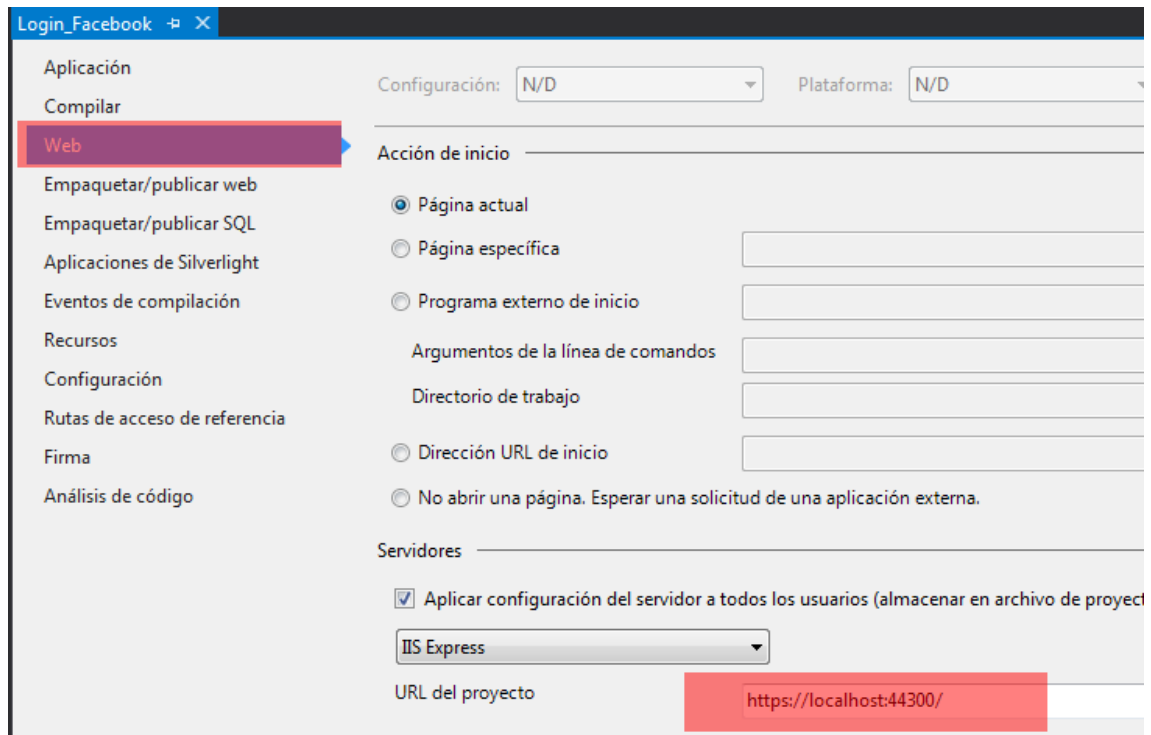


Para conectar a la autenticación de proveedores como Google y Facebook, tendremos que configurar IIS-Express para utilizar SSL. Es importante seguir usando SSL después de inicio de sesión y no caer de nuevo a HTTP, la cookie de conexión es tan secreta como el nombre de usuario y contraseña, y sin utilizar SSL se va a enviar en texto sin cifrar.

Ingresamos a propiedades del proyecto presionando F4 sobre este. Habilitaremos SSL y copiaremos la dirección URL de SSL (esta URL será <https://localhost:44300/> a menos que previamente se hayan creado otros proyecto SSL)



Ingresamos a las propiedades del proyecto y reemplazamos la dirección URL de SSL.

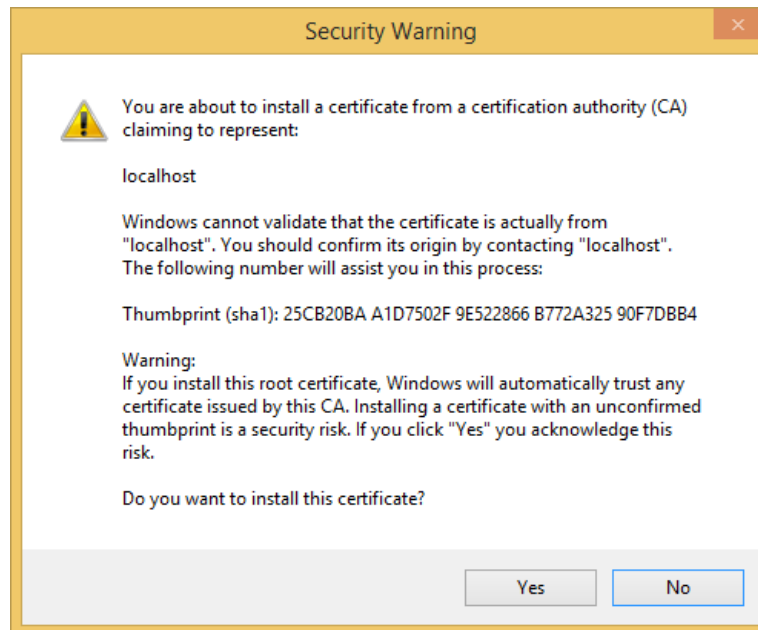


Añadir el atributo `RequireHttps` al controlador principal para exigir a todas las solicitudes que deben utilizar HTTPS.

```
[RequireHttps]
public class HomeController : Controller
{
    public ActionResult Index()
    {
        return View();
    }
}
```

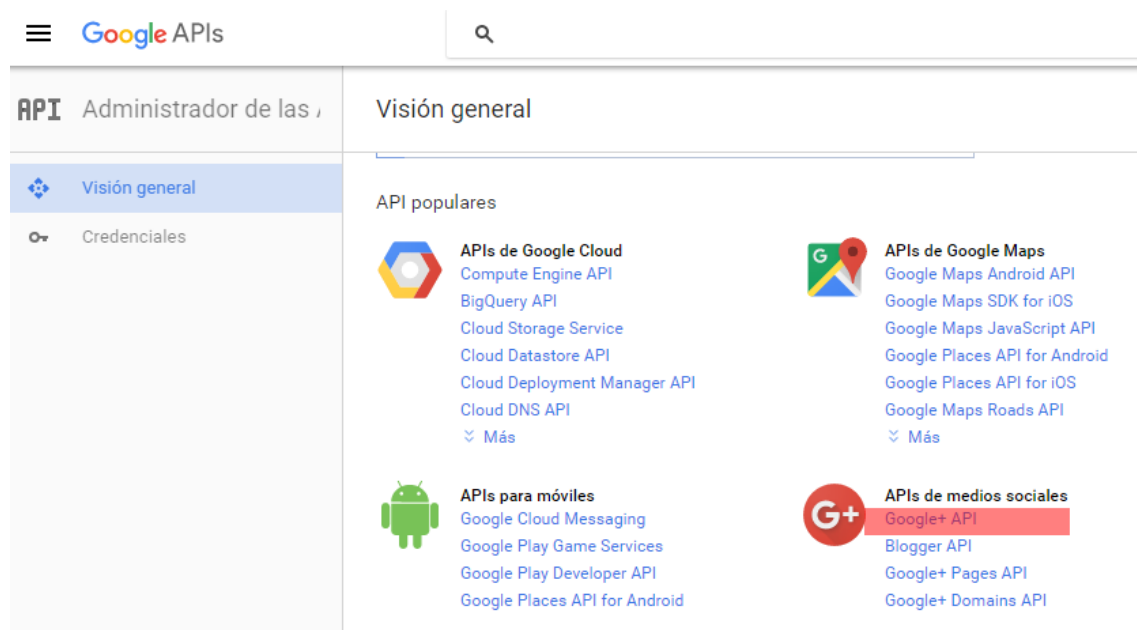
Compilamos la aplicación y aceptamos lo siguiente.



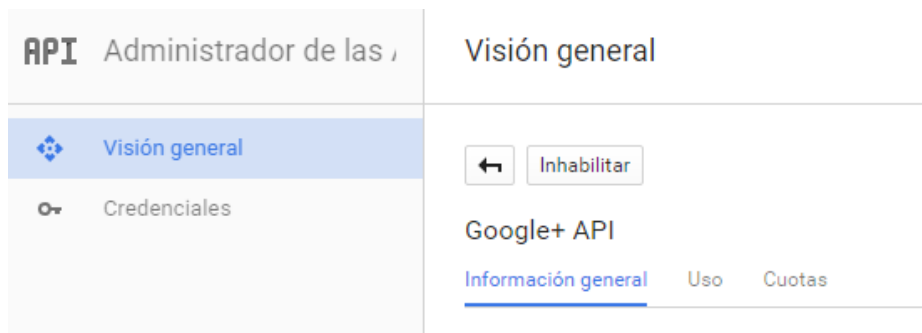


HABILITANDO REGISTRO CON GOOGLE

Ingresamos a la siguiente URL <https://console.developers.google.com> nos registramos y seleccionamos la opción Google + API.



Debemos habilitar Google + API



A continuación creamos un proyecto, lo nombramos de la manera que gustemos

Visión general

⚠ Se necesita un proyecto para habilitar las API [Crear proyecto](#)

⬅ [Habilitar](#)

Crear proyecto

Google Developers Console utiliza proyectos para gestionar recursos. Para empezar, crea el primer proyecto.

Nombre del proyecto ?

My Project

ID del proyecto ?

indigo-bloom-130602

[Mostrar las opciones avanzadas...](#)

Quiero recibir por correo electrónico información sobre las funciones del producto, sugerencias de rendimiento, encuestas para aportar mis observaciones y ofertas especiales.

☐ Sí ☒ No

Acepto que el uso que haga de cualquier [servicio y API relacionadas](#) queda sujeto a mi cumplimiento de las [Condiciones de Servicio](#) correspondientes.

☒ Sí ☐ No

[Crear](#)

Luego debemos ir a la opción de credenciales y crear una credencial del tipo “ID de cliente de OAuth”

API

Administrador de las

Credenciales

Visión general

Credenciales

Credenciales

Pantalla de autorización de OAuth

Verificación de dominio

Crear credenciales

Eliminar

Clave de API

Identifica el proyecto mediante una simple clave de API para comprobar la cuota y acceso.
Para APIs como la del Traductor de Google.

ID de cliente de OAuth

Solicita la autorización del usuario para que la aplicación pueda acceder a los datos del usuario.
Se puede utilizar con APIs como la de Google Calendar.

Clave de cuenta de servicio

Habilita la autenticación de servidor a servidor en el nivel de aplicación mediante cuentas robot.
Se puede utilizar con las API de Google Cloud.

Ayúdame a elegir

Sus preguntas te ayudarán a decidir el tipo de credencial que quieres usar.

Debemos ingresar <https://localhost:44300> y <https://localhost:44300/signin-google> en el siguiente formulario

Tipo de aplicación

☒ Web
 ☐ Android [Más información](#)
☐ Chrome [Más información](#)
☐ iOS [Más información](#)
☐ PlayStation 4
 ☐ Otro

Nombre

Restricciones

Introduce los orígenes de JavaScript, los URI de redireccionamiento o ambos

Orígenes de JavaScript autorizados

Para su uso en las solicitudes de navegador. Se trata del URI de origen de la aplicación cliente. No puede contener caracteres comodín (http://*.example.com) ni una ruta (<http://example.com/subdir>). Si utilizas un puerto no estándar, deberás incluirlo en el URI de origen.

URIs de redireccionamiento autorizados

Para usarse con las solicitudes de un servidor web. Es la ruta de la aplicación a la que se redirecciona a los usuarios después de autenticarse en Google. A dicha ruta se añadirá el código de autorización de acceso. Debe tener un protocolo. No puede incluir fragmentos de URL ni rutas relativas. No puede ser una dirección IP pública.

Nos saldrá una ventana con los datos del ID de cliente y secreto de cliente

Cliente de OAuth

Este es tu ID de cliente

143136686448-e8b9bjeg5si9gfr7u9ga2l3ob2qp70c.apps.googleusercontent.com




Este es tu secreto de cliente

Q6YVbDEdOGZ7-Pg0J-_K2b5e

Aceptar

En caso no haber copiado los datos anteriores, podemos descargar el JSON de la siguiente manera

IDs de cliente de OAuth 2.0

<input type="checkbox"/> Nombre	<input type="text" value="Fecha de creación"/>	<input type="text" value="Tipo"/>	<input type="text" value="ID de cliente"/>	
<input type="checkbox"/> Cliente web 2	8 may. 2016	Web	143136686448-e8b9bjeg5si9gfr7u9ga2l3ob2qp70c.apps.googleusercontent.com	 
<input type="checkbox"/> Cliente web 1	8 may. 2016	Web	143136686448-4nvvi5v15ouk6ulhnb36sepc5ifctnb0.apps.googleusercontent.com	

Y copiamos los siguientes datos.

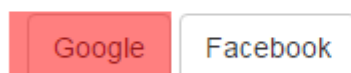
```
client_secret_1431...sercontent.com.json* X ManageController.cs HomeController.cs
<No se seleccionó ningún esquema>
{
  "web": {
    "client_id": "143136686448-e8b9bjeg5si9gfr7u9ga2l3ob2qp70c.apps.googleusercontent.com",
    "project_id": "gothic-agility-130623",
    "auth_uri": "https://accounts.google.com/o/oauth2/auth",
    "token_uri": "https://accounts.google.com/o/oauth2/token",
    "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
    "client_secret": "Q6YVbDEdOGZ7-Pg0J-_K2b5e",
    "redirect_uris": [ "https://localhost:44300/signin-google" ],
    "javascript_origins": [ "https://localhost:44300" ] }
}
```

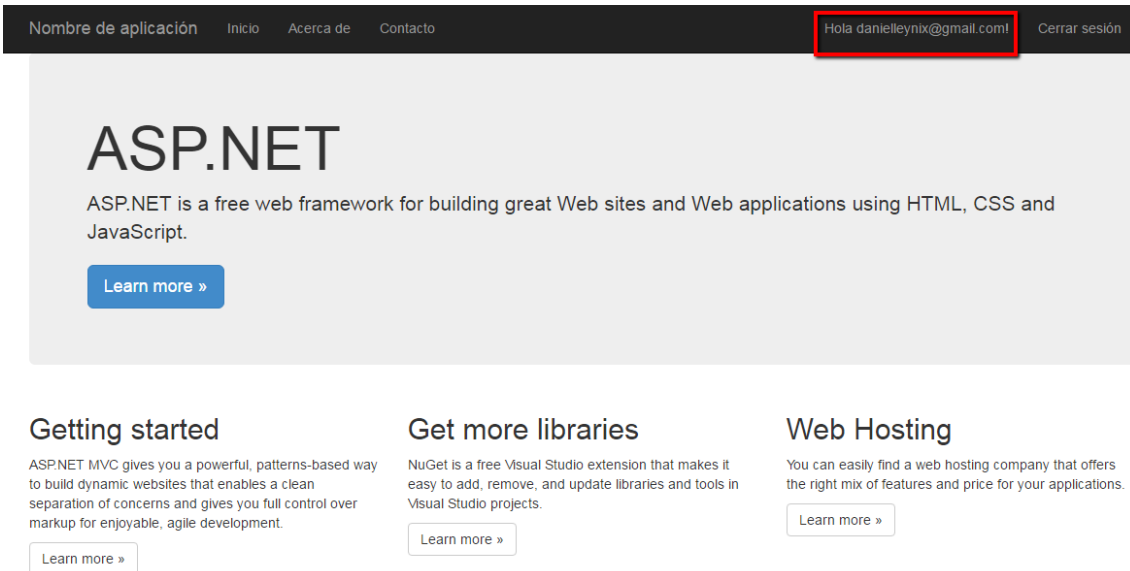
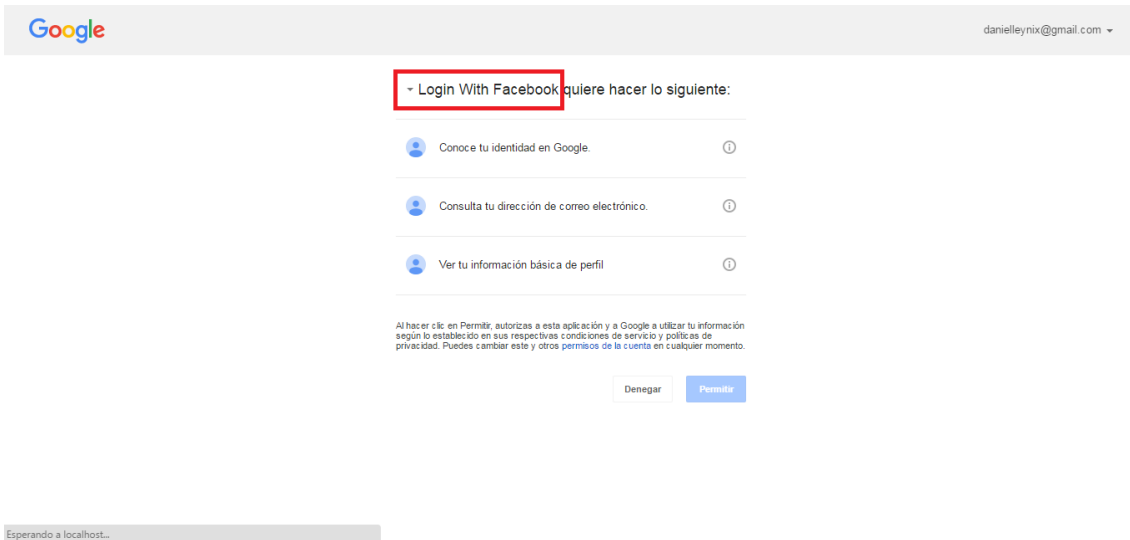
Ingresamos a App_Start/Startup.Auth.cs y dentro del método UseGoogleAuthentication ingresamos los parámetros recibidos previamente

```
app.UseGoogleAuthentication(new GoogleOAuth2AuthenticationOptions()
{
    ClientId = "143136686448-4nvvi5v15ouk6ulhnb36sepc5ifctnb0.apps.googleusercontent.com",
    ClientSecret = "UHxv_yMum1SLMhShqx9AYnig"
});
```

Ejecutamos el proyecto y al iniciar sesión utilizamos el servicio de Google

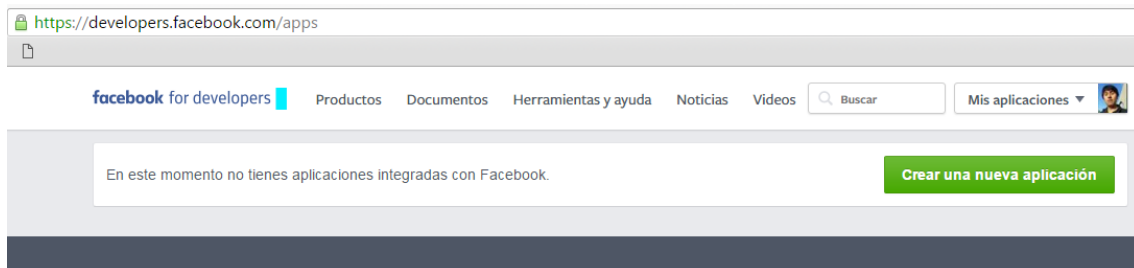
Use otro servicio para iniciar sesión.





HABILITANDO EL REGISTRO CON FACEBOOK

Ingresamos a <https://developers.facebook.com/> nos registramos y creamos una nueva aplicación



La nombramos y seleccionamos la categoría “Aplicaciones para páginas”

Crear un nuevo identificador de la aplicación

Empezar a integrar Facebook en tu aplicación o sitio web

Nombre para mostrar

ASP Ident

Espacio de nombres

Identificador único de tu aplicación (opcional)

Correo electrónico de contacto

Se usa para comunicaciones importantes sobre tu aplicación

Categoría

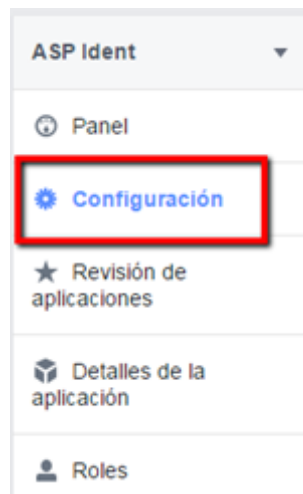
Aplicaciones para páginas ▼

Al continuar, aceptas las Políticas de la plataforma de Facebook

Cancelar

Crear identificador de la aplicación




Luego ingresamos a la sección de configuración



Ya se nos generan los datos que necesitaremos desde nuestra web. Agregamos una nueva plataforma que en esta ocasión será una Web

Básica	Avanzada
Identificador de la aplicación 120796648329990	Clave secreta de la aplicación Mostrar
Nombre para mostrar ASP Ident	Espacio de nombres
Dominios de la aplicación	Correo electrónico de contacto danielleynix@gmail.com
<div>+ Agregar plataforma</div>	
<div>Descartar Guardar cambios</div>	

Selecciona la plataforma

 Canvas de Facebook	 Sitio web	 iOS	 Android
 Aplicación para Windows	 Pestaña de la página	 Xbox	 Play Station

Copiamos los datos referentes a “Identificador de la aplicación” y “Clave secreta de la aplicación” que serán nuestros parámetros a usar.

Básica	Avanzada
Identificador de la aplicación <input type="text" value="120796648329990"/>	Clave secreta de la aplicación <input type="password" value="••••••••"/> <input type="button" value="Mostrar"/>
Nombre para mostrar <input type="text" value="ASP Ident"/>	Espacio de nombres <input type="text"/>
Dominios de la aplicación <input type="text"/>	Correo electrónico de contacto <input type="text" value="danielleynix@gmail.com"/>
Sitio web <input type="button" value="Inicio rápido"/> <input type="button" value="✕"/>	
URL del sitio <input type="text" value="https://localhost:44300/"/>	
+ Agregar plataforma	
<input type="button" value="Descartar"/> <input type="button" value="Guardar cambios"/>	

Avanzada

Identificador de la aplicación

120796648329990

Clave secreta de la aplicación

●●●●●●●

Mostrar

Nombre para mostrar

ASP Ident

Espacio de nombres

Dominios de la aplicación

Correo electrónico de contacto

danielleynix@gmail.com

Sitio web

Inicio rápido

URL del sitio

<https://localhost:44300/>

+ Agregar plataforma

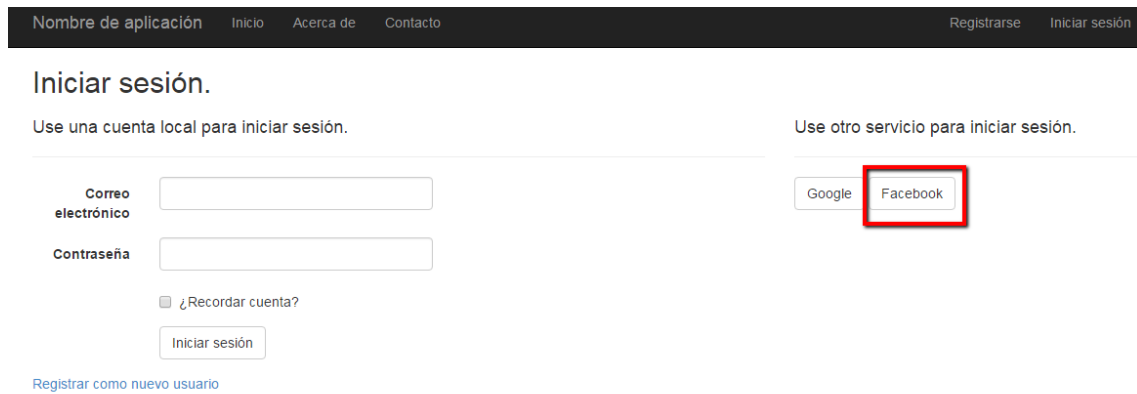
Descartar

Guardar cambios

Ingresamos a App_Start/Startup.Auth.cs y dentro del método UseFacebookAuthentication ingresamos los parámetros recibidos previamente obtenidos

```
app.UseFacebookAuthentication(
    appId: "120796648329990",
    appSecret: "eb7f7538839730b7ad6fc543c851d5ff");
```

Ejecutamos el proyecto y seleccionamos la opción de Facebook



Nombre de aplicación Inicio Acerca de Contacto Registrarse Iniciar sesión

Iniciar sesión.

Use una cuenta local para iniciar sesión.

Correo electrónico

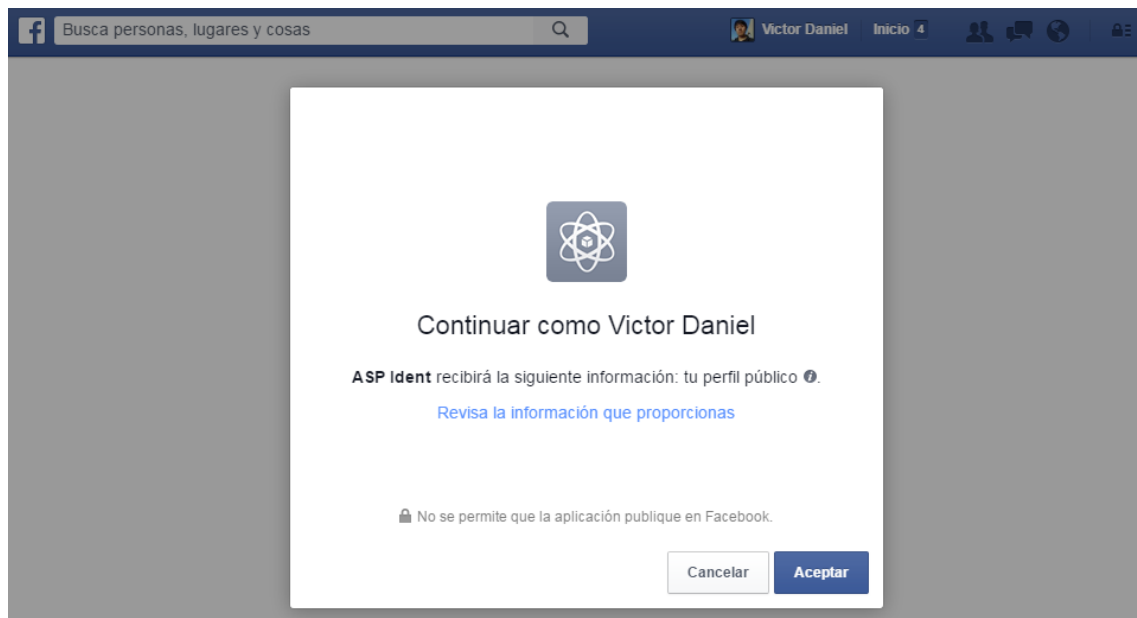
Contraseña

☐ ¿Recordar cuenta?

[Registrar como nuevo usuario](#)

Use otro servicio para iniciar sesión.

Nos registramos y aceptamos.



Busca personas, lugares y cosas

Victor Daniel Inicio 4

Continuar como Victor Daniel

ASP Ident recibirá la siguiente información: tu perfil público ⓘ

[Revisa la información que proporcionas](#)

🔒 No se permite que la aplicación publique en Facebook.

El registro se completa al ingresar un correo electrónico para usar en la web.

Registrarse.

Asocie su cuenta Facebook.

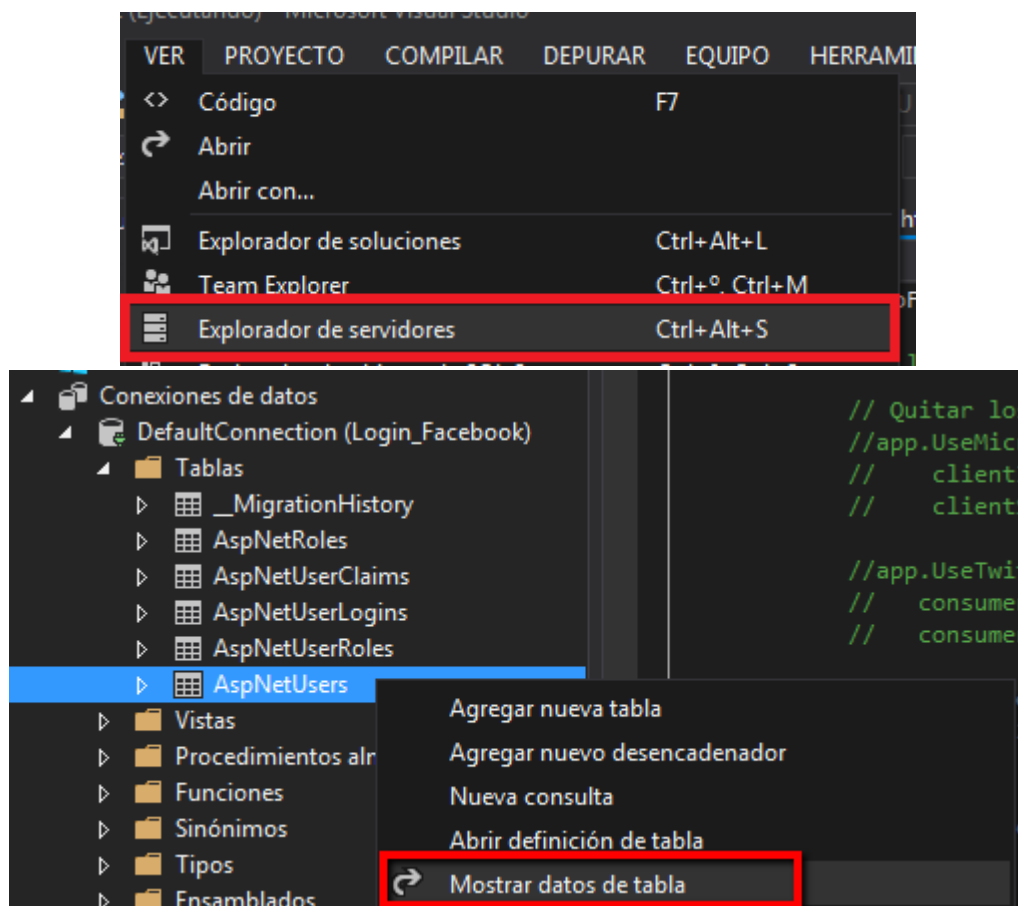
Formulario de asociación

Se autenticó correctamente con Facebook. Introduzca un nombre de usuario para este sitio y haga clic en el botón Registrar para finalizar el inicio de sesión.

Correo electrónico

DATOS ALMACENADOS

Para ver los datos que vamos almacenando los hacemos de la siguiente manera



dbo.AspNetUsers [Datos] | Startup.Auth.cs | Login.cshtml

Máximo de filas: 1000

	Id	Email	EmailConfirmed	PasswordHash	SecurityStamp	PhoneNumber	PhoneNumber...	TwoFactorEna...	LockoutEnd...
	441f34f7-095d-4dab-a...	victor_14_nyx@...	False	NULL	bc02300b-7dba...	NULL	False	False	NULL
	c8602524-fee4-fb1-b...	danielleynix@g...	False	NULL	c7bb4e15-260e...	NULL	False	False	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

CAPTURANDO NUEVOS ATRIBUTOS DEL CLIENTE

Ingresamos a Models/IdentityModels.cs e ingresamos los siguientes datos, para almacenar más datos del cliente. El mismo cambio realizamos en Models/AccountViewModels.cs

```
namespace Login_Facebook.Models
{
    public class ExternalLoginConfirmationViewModel
    {
        [Required]
        [Display(Name = "Correo electrónico")]
        public string Email { get; set; }

        public string HomeTown { get; set; }
        public System.DateTime? BirthDate { get; set; }
    }
}
```

```
namespace Login_Facebook.Models
{
    // Puede agregar datos del perfil del usuario agregando
    public class ApplicationUser : IdentityUser
    {
        public string HomeTown { get; set; }
        public System.DateTime? BirthDate { get; set; }
    }
}
```

Ingresamos a Controllers\AccountController.cs y añadimos los nuevos campos en el método ExternalLoginConfirmation.

```
//
// POST: /Account/ExternalLoginConfirmation
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<ActionResult> ExternalLoginConfirmation(ExternalLoginConfirmationViewModel model, string returnUrl)
{
    if (User.Identity.IsAuthenticated)
    {
        return RedirectToAction("Index", "Manage");
    }

    if (ModelState.IsValid)
    {
        // Obtener datos del usuario del proveedor de inicio de sesión externo
        var info = await AuthenticationManager.GetExternalLoginInfoAsync();
        if (info == null)
        {
            return View("ExternalLoginFailure");
        }
        var user = new ApplicationUser { UserName = model.Email, Email = model.Email,
            BirthDate = model.BirthDate,
            HomeTown = model.HomeTown
        };
        var result = await UserManager.CreateAsync(user);
    }
}
```

Ingresamos a Views/Account/ExternalLoginConfirmation.cshtml y adaptamos la vista.

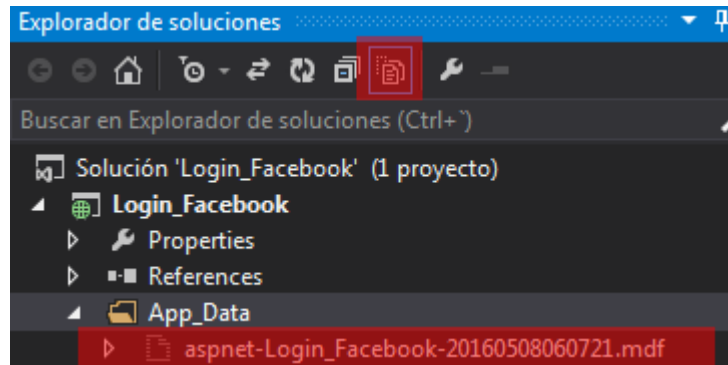
```
ExternalLoginConfirmation.cshtml AccountController.cs AccountViewModels.cs IdentityModels.cs

<p class="text-info">
    Se autenticó correctamente con <strong>@ViewBag.LoginProvider</strong>.
    Introduzca un nombre de usuario para este sitio y haga clic en el botón Registrar para
    el inicio de sesión.
</p>

<div class="form-group">
    @Html.LabelFor(m => m.Email, new { @class = "col-md-2 control-label" })
    <div class="col-md-10">
        @Html.TextBoxFor(m => m.Email, new { @class = "form-control" })
        @Html.ValidationMessageFor(m => m.Email, "", new { @class = "text-danger" })
    </div>
</div>

<!--Nuevos datos-->
<div class="form-group">
    @Html.LabelFor(m => m.HomeTown, new { @class = "col-md-2 control-label" })
    <div class="col-md-10">
        @Html.TextBoxFor(m => m.HomeTown, new { @class = "form-control" })
        @Html.ValidationMessageFor(m => m.HomeTown)
    </div>
</div>
<div class="form-group">
    @Html.LabelFor(m => m.BirthDate, new { @class = "col-md-2 control-label" })
    <div class="col-md-10">
        @Html.TextBoxFor(m => m.BirthDate, new { @class = "form-control" })
        @Html.ValidationMessageFor(m => m.BirthDate)
    </div>
</div>
<!--/Nuevos datos-->
```

En el explorador de soluciones seleccionamos la opción “Mostrar todos los archivos” y borramos el archivo señalado que está ubicado en “App_Data” para eliminar la base de datos “membership” y poder registrar los datos nuevamente, pero esta vez capturando los nuevos atributos.



El registro con Google sería de la siguiente forma

Use otro servicio para iniciar sesión.

Google

Facebook

Registrarse.

Asocie su cuenta Google.

Formulario de asociación

Se autenticó correctamente con **Google**. Introduzca un nombre de usuario para este sitio y haga clic en el botón Registrar para finalizar el inicio de sesión.

Correo electrónico	<input type="text" value="danielleynix@gmail.com"/>
HomeTown	<input type="text" value="Lima"/>
BirthDate	<input type="text" value="07/10/1995"/>

Registrarse

El registro con Facebook sería de la siguiente forma

Use otro servicio para iniciar sesión.

Google

Facebook

Registrarse.

Asocie su cuenta Facebook.

Formulario de asociación

Se autenticó correctamente con **Facebook**. Introduzca un nombre de usuario para este sitio y haga clic en el botón Registrar para finalizar el inicio de sesión.

Correo electrónico	<input type="text" value="victor_14_nyx@gmail.com"/>
HomeTown	<input type="text" value="Lima"/>
BirthDate	<input type="text" value="07/10/1995"/>
<input type="button" value="Registrarse"/>	

DATOS CAPTURADOS DEL REGISTRO

El método "AuthenticationManager.GetExternalLoginInfoAsync()" es el que captura los datos de la sesión cuando se registra con Google o Facebook

```
AuthenticationManager.GetExternalLoginInfoAsync();
```

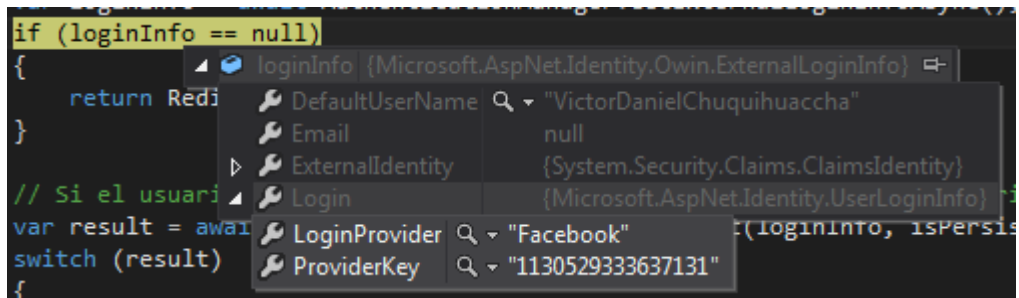
Al registrar con Google se obtiene los siguientes datos

```
//  
// GET: /Account/ExternalLoginCallback  
[AllowAnonymous]  
public async Task<ActionResult> ExternalLoginCallback(string returnUrl)  
{  
    var loginInfo = await AuthenticationManager.GetExternalLoginInfoAsync();  
    if (loginInfo == null)  
    {  
        return RedirectToAction("Register");  
    }  
    // Si el usuario ya tiene un inicio de sesión, iniciar sesión del usuario  
    var result = await SignInManager.ExternalSignInAsync(loginInfo, isPersistent: false);  
    switch (result)  
    {  
        case SignInResult.Success:  
            return RedirectToAction("Index", "Home");  
        case SignInResult.Unknown:  
            return RedirectToAction("Register");  
        case SignInResult.Failure:  
            return RedirectToAction("Register");  
    }  
}
```

```
if (loginInfo == null)  
{  
    return RedirectToAction("Register");  
}  
// Si el usuario ya tiene un inicio de sesión, iniciar sesión del usuario  
var result = await SignInManager.ExternalSignInAsync(loginInfo, isPersistent: false);  
switch (result)  
{  
    case SignInResult.Success:  
        return RedirectToAction("Index", "Home");  
    case SignInResult.Unknown:  
        return RedirectToAction("Register");  
    case SignInResult.Failure:  
        return RedirectToAction("Register");  
}
```

Al registrar con Facebook se obtiene los siguientes datos

```
if (loginInfo == null)  
{  
    return RedirectToAction("Register");  
}  
// Si el usuario ya tiene un inicio de sesión, iniciar sesión del usuario  
var result = await SignInManager.ExternalSignInAsync(loginInfo, isPersistent: false);  
switch (result)  
{  
    case SignInResult.Success:  
        return RedirectToAction("Index", "Home");  
    case SignInResult.Unknown:  
        return RedirectToAction("Register");  
    case SignInResult.Failure:  
        return RedirectToAction("Register");  
}
```

REFERENCIAS

- ✓ <http://www.asp.net/mvc/overview/security/create-an-aspnet-mvc-5-app-with-facebook-and-google-oauth2-and-openid-sign-on#goog>